NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

SIMULATION AND ANALYSIS OF A WIRELESS MAC PROTOCOL: MACAW

by

Tufan Oruk

September 1996

Thesis Advisor: Gilbert M. Lundy
Second Reader: Man-Tak Shing

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 1996	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE SIMULATION AND ANALYSIS OF A WIRELESS MAC PROTOCOL: MACAW

5. FUNDING NUMBERS

6. AUTHOR(S) Tufan Oruk

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

MACAW (Medium Access Collision Avoidance Wireless) is a new MAC protocol for wireless LANs proposed by Bharghavan et al. [Ref. 2] based on Karn's MACA protocol [Ref.3]. In this thesis the performance characteristics and operational behavior of the protocol are investigated.

The approach taken was to simulate the protocol by OPNET 2.4c of MIL3, Inc. and determine the utilizations and mean delay times of the transmitters under various operational conditions. Also a new performance measure was defined in terms of utilization and mean delay time.

Our investigation has shown that the optimum performance of the MACAW protocol occurs at approximately 50% channel load. We have also shown the importance of the backoff algorithm, and finally, we have shown that carrier sensing dramatically improves the performance of the protocol for high channel loads.

Simulation results showed that decreasing the backoff increase rate by 15% gave twice as good performance results for the small number of transmitting nodes cases. When carrier sensing was introduced to the protocol, dramatic performance increases resulted under heavy loads (60% to 80% channel loads). Carrier sensing also pushed the optimum performance channel load threshold from 50% to 60%.

14. SUBJECT TERMS Wireless, LAN, Simulation, Analysis, Protocol, Formal Analysis			15. NUMBER OF PAGES 86
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

Approved for public release; distribution is unlimited.

SIMULATION AND ANALYSIS OF A WIRELESS MAC PROTOCOL: MACAW

Tufan Oruk
Lieutenant Junior Grade, Turkish Navy
B.S. Turkish Naval Academy, 1990

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE
IN
COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1996

ABSTRACT

MACAW (Medium Access Collision Avoidance Wireless) is a new MAC protocol for wireless LANs proposed by Bharghavan et al. [Ref. 2] based on Karn's MACA protocol [Ref.3]. In this thesis the performance characteristics and operational behavior of the protocol are investigated.

The approach taken was to simulate the protocol by OPNET 2.4c of MIL3, Inc. and determine the utilizations and mean delay times of the transmitters under various operational conditions. Also a new performance measure was defined in terms of utilization and mean delay time.

Our investigation has shown that the optimum performance of the MACAW protocol occurs at approximately 50% channel load. We have also shown the importance of the backoff algorithm, and finally, we have shown that carrier sensing dramatically improves the performance of the protocol for high channel loads.

Simulation results showed that decreasing the backoff increase rate by 15% gave twice as good performance results for the small number of transmitting nodes cases. When carrier sensing was introduced to the protocol, dramatic performance increases resulted under heavy loads (60% to 80% channel loads). Carrier sensing also pushed the optimum performance channel load threshold from 50% to 60%.

TABLE OF CONTENTS

I. INTRODUCTION

After the first practical use of wireless communications in 1895 by Marconi, radio communications rapidly evolved and recent advances enabled wireless technology to be used in computers effectively and economically, even when the computers are mobile. Today wireless/mobile communication is in big growth worldwide (Figure 1). Most of the portable computers (notebooks, PDAs etc.) come with built in wireless functionalities such as infrared, paging, cellular modem. Every day networked wireless computers are becoming a greater part of the computing infrastructure. Since Local Area Networks (LAN) are a big part of this infrastructure they also take part in this technological development.

Figure 1. Wireless LAN Revenues
(From BIS Strategic Decisions)

Mobility is the major advantage that wireless technology provides. Wireless LANs can be built quickly and effectively used in locations where wiring is difficult such as historical buildings, factories and in temporary sites such as trade shows and disaster areas.

There are three primary application areas for wireless LAN technology [Ref. 1]

- Wireless desktops
- Wireless building-to-building short-haul communications
- Portable mobile connectivity applications

There are many wireless data technologies. Some of these are [Ref. 1]:

- Spread Spectrum
- Infrared
- Mobile Radio
- Meteor Burst Communication
- Cellular telephony
- Microwave
- Very Small Aperture Terminal (VSAT)
- Mobile Satellite Communications
- FM squared
- FM sideband
- Packet Radio

In this thesis we are interested in packet radio single channel medium access control protocols for wireless LANs. Single channel medium access protocols can be categorized into two groups [Ref. 2]

- Multiple access
- Token based

We will investigate performance characteristics of a new multiple access wireless MAC protocol called MACAW (Multiple Access Collision Avoidance Wireless), proposed by Bharghavan et al [Ref. 2].

The organization of the thesis is as follows. In Chapter II MACAW protocol overview is given. In Chapter III simulation program OPNET 2.4c from MIL3, Inc is introduced. In Chapter IV test cases that will be simulated are discussed. In Chapter V a performance measure that will be used for comparisons is introduced, some definitions and values regarding the simulations are given and the results of the simulations are summarized. In Chapter VI some modifications are proposed to the original protocol and simulation results of these modified protocols are given. Finally, in Chapter VII all of the findings are summarized.

II. MACAW PROTOCOL SUMMARY

MACAW [Ref. 2] was developed at Xerox PARC based on the Multiple Access Collision Avoidance (MACA) protocol proposed by Karn [Ref. 3]. Like MACA, MACAW is a single channel, multiple access wireless local area network protocol. Developers of MACAW have added new features to the MACA protocol, based on the observations that they have made at the Computer Science Laboratory of Xerox PARC. The differences between MACA and MACAW are as follows:

- Use of RTS-CTS-DS-DATA-ACK message exchange, instead of RTS-CTS-DATA message exchange sequence.
- Per "stream" basis "multiplicative increase linear decrease" backoff algorithm, instead of per node basis "binary exponential" backoff algorithm.
- Distribution of congestion information via a "backoff copying" schema.

These modifications are based on the following observations that the developers had made:

- Contention is at the receiver, so carrier sensing is inappropriate.
- Congestion is location dependent (at the receiver). This observation follows from the previous one.
- To allocate media fairly, local congestion information must be known by all nodes. That is, congestion information propagates through the network. (This is done by backoff copying algorithm).
- To provide effective contention among nodes, protocol should propagate synchronization information.

There are two types of packets in the MACAW protocol. Control packets (RTS-CTS-DS-ACK-RRTS) and data packet (DATA).

All control packets are 30 bytes long. A "slot" is the time that a control packet is transmitted. As long as this time slot is greater than the round trip time from the nodes in the

"vicinity," after a successful RTS-CTS exchange, there would not be a collision with DATA packets, under the assumption that there are no hidden nodes[Ref. 4].

RTS	Request to Send
CTS	Clear to Send
DS	Data Send
ACK	Acknowledge
RRTS	Request to RTS
DATA	Data

Table 1. Packets in MACAW Protocol

Transmission Speed	(for Slot = 30 bytes) Slot length [m]
1 Kbps	70,312,500.0
10 Kbps	7,031,250.0
100 Kbps	703,125.0
1000 Kbps	70,312.5

Table 2. Thirty Byte "Slot" Length with Different Transmission Speeds

In [Ref. 2] 8 states are defined for MACAW protocol. The state analysis of this protocol [Ref. 5] by using Systems of Communicating Machines model [Ref. 6] proposed that 10 states are needed to resolve some ambiguities. The original states are IDLE, CONTEND, WFCTS (wait for CTS), WFACK, WFDS, WFDATA, QUIET and WFCONTEND and two new states [Ref. 5] are XMIT and CONTEND_2. These new states do not change the operation of the protocol at all, but resolves some ambiguities.

There is no explicit packet format mentioned in the original paper [Ref. 2]. But we can infer the packet format easily by considering what information is to be sent to the other node.

A. OPERATION OF THE PROTOCOL

Protocol will be described as four sets of rules [Ref. 2].

- Backoff algorithm rules
- Control rules

- Deferral rules
- Timeout rules

Control Packet

SA	DA	TYPE	DATA_SIZE	SQN	LOCAL_BO	REMOTE_BO

Data Packet

SA	DA	TYPE	DATA_SIZE	SQN	LOCAL_BO	REMOTE_BO
DATA						

SA	: Source Address
DA	: Destination Address
TYPE	: Packet Type (one of RTS, CTS, DS, DATA, ACK, RRTS)
DATA_SIZE	: Size of Data that is Sent in the DATA Packet (in Bytes)
LOCAL_BO	: Local Backoff
REMOTE_BO	: Remote Backoff
SQN	: Sequence Number of Data Packet
DATA	: Data of the Data Packet

Figure 2. Control and Data Packet Formats

1. Backoff Algorithm Rules

MACAW uses a very different backoff algorithm than MACA. The goal is to distribute local congestion information through the network and provide fair channel access to the streams rather than to the nodes. A stream is a data flow from a source to a destination. A node can have many streams. In the packet format there are two fields that hold this information. These fields are "local backoff" and "remote backoff."

The backoff algorithm that is used in the simulations is a slightly simplified version. Retry count for streams are not used to calculate new backoff value as described in [Ref. 2] "Backoff and Copying Rules," but multiplicative increase algorithm is used as described in the discussion of backoff algorithm in the same paper. These do not have any significant difference from each other to a range up to 64, which is the maximum backoff value for the network, chosen by the developers. Not using retry count for the streams decreased the bookkeeping in the implementation of the protocol.

Every node has two backoff tables. These tables hold the "local" and "remote" backoff information of the corresponding streams. Besides these tables, each node keeps track of its latest backoff value. Following variables are belong to node Q.

- Q.local_backoff[R] : the backoff value of this station as estimated by the remote station R.
- Q.remote_backoff[R]: the estimated backoff value for the remote station R by this station
- Q.my_backoff: this node's latest backoff value

Backoff information is exchanged between nodes according to the following rules.

Case 1: P wants to send a packet to Q. P sets backoff fields of the packet in the following way.

```
if(packet.type == RTS){
packet.local_backoff = my_backoff;
} else {
        packet.local_backoff = local_backoff[Q];
}
packet.remote_backoff = remote_backoff[Q];
```

Case 2: Q receives a packet from P that is addressed to itself. Then Q updates its backoff tables in the following way.

```
/* local_backoff[Q] of P is remote_backoff[P] of Q
* Q decides that whether the incoming packet is retransmit
* or not by looking at its sequence number
*/
if(this_is_not_a_retransmit){
        remote_backoff[P] = packet.local_backoff;
        local_backoff[P]  = packet.remote_backoff;
}
/* this is a retransmission caused by
* a congestion at the other side
* So Q will increase the remote_backoff value for P 1.5 times
*/
else {
        remote_backoff[P] = (remote_backoff[P] * 1.5 > BO_MAX ?
                                        BO_MAX : remote_back_off[P] * 1.5);
        /* to keep corresponding backoff values between nodes in
         * balance make their local and remote backoff totals be
         * equal
         */
        local_backoff[P] = (packet.local_backoff +
                        packet.remote_backoff) - remote_backoff[P];
}
```

BO_MAX is the maximum backoff value allowed in the network. This is determined before setting up the network. Backoff value is increased multiplicatively (1.5 times in this case). This is the multiplicative increase portion of the backoff algorithm. This provides gentler increase in the backoff values than the binary exponential backoff algorithm.

Case 3: R hears a packet from P to Q. R updates its backoff tables in the following way.

```
if(packet.type != RTS){
                remote_backoff[P] = packet.local_backoff;
                remote_backoff[Q] = packet.remote_backoff;
}
```

R does not update its tables when it hears an RTS message.

This schema helps nodes to keep track of the congestion for each of its streams. When P wants to send a packet to Q it will defer by using the remote backoff information of the Q (remote_backoff[Q]). And P will inform other nodes about its local congestion (with respect to Q) by using its local_backoff[Q] information (in RTS packet it uses my_backoff value). Since every node knows each other's local congestion information (with respect to itself), they

employ better deferral times and they should cause less collisions. The result should be better utilization of the channel.

2. Control Rules

As mentioned above, a node running MACAW protocol can be in one of ten states (Figure 3. MACAW Protocol State Machine). IDLE, CONTEND, WFCTS, XMIT, WFACK, QUIET, WFCONTEND, CONTEND_2, WFDS, and WFDATA. A node should receive all of the packet to understand what type it is and should send all the packet before it sets its defer timer. With these in mind, transition rules are as follows:

1. A station begins to run the MACAW protocol from IDLE state.

2. When node A is in IDLE state and has data to transmit (either new data or a retransmit) to node B, it set its defer_timer to a random value which is determined by using remote backoff value of B, and goes to CONTEND state.

```
defer_timer = TIME_SLOT * rand(BO_MIN...remote_backoff[B]);
```

3. When A is in CONTEND state and its defer_timer expires, it transmits an RTS to B, sets its timer enough to receive CTS and goes to WFCTS state.

```
defer_timer = TIME_SLOT + ROUND_TRIP_TIME
```

4. When B is in IDLE state and receives an RTS from A , B immediately transmits a CTS to A, sets its defer_timer enough to receive DS packet and goes to WFDS state.

```
defer_timer = TIME_SLOT + ROUND_TRIP_TIME;
```

5. When A is in WFCTS state and receives a CTS packet form B, it clears its defer_timer and transmits DS and DATA packets back-to-back. When transmissions are over sets its timer enough to receive ACK from B and goes to WFACK state.

```
defer_timer = TIME_SLOT + ROUND_TRIP_TIME;
```

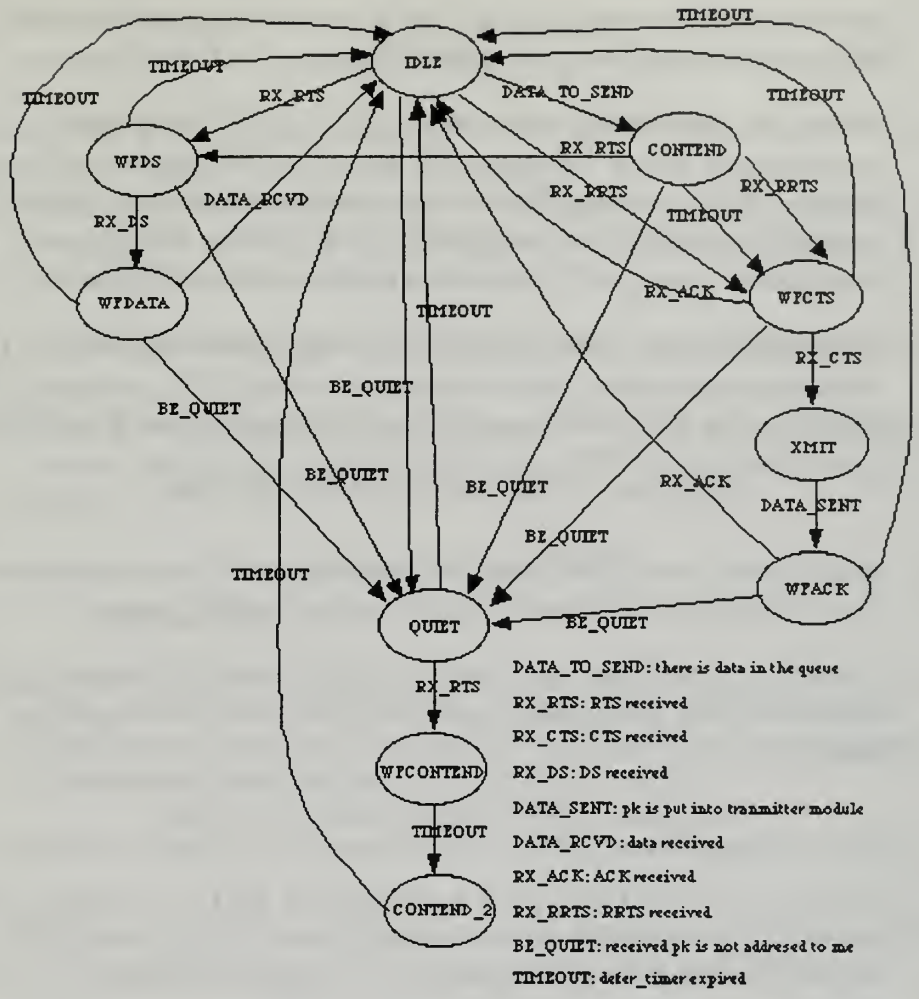


Figure 3. MACAW Protocol State Machine

6. When B is in WFDS state and receives DS packet from A, it goes to WFDATA state and sets its defer_timer enough to receive all data packet.

```
defer_timer = node_transmit_speed / sent_data_size; // [bps / bits]
```

7. When B is in WFDATA state and receives the data packet from A , B transmits an ACK packet to A and goes back to IDLE state.

8. When A is in WFACK state and receives an ACK from B, it resets its defer_timer and goes back to IDLE state.

9. When B is IDLE state and receives an RTS for a packet that has been acknowledged before, it increases the remote backoff value of the sending station

assuming that there is a congestion at that station's side and retransmits an ACK to it. B does not change its state, stays in IDLE.

10. When B is CONTEND state and receives an RTS for a packet that has been acknowledged before, it increases the remote backoff value of this sending station assuming there is a congestion at its side and retransmits an ACK to it. B keeps on waiting to send its "contending" RTS. This behavior is not present in [Ref. 2] control rules section but it is implied in the discussion of ACK packet.

11. When A receives an RTS packet for a new data packet while it is in CONTEND state, A resets its defer_timer which was set to send RTS, transmits a CTS to the sender, sets its defer_timer enough to receive DS packet and goes to WFDS state.

```
defer_timer = TIME_SLOT + ROUND_TRIP_TIME;
```

12. When node C is in QUIET state and receives an RTS that is addressed to itself, it goes to WFCONTEND state (without resetting its defer_timer).

13. When D is in IDLE state and receives an RRTS packet, D transmits an RTS to the sender of RRTS, sets its defer_timer enough to receive CTS and goes to WFCTS state.

```
defer_timer = TIME_SLOT + ROUND_TRIP_TIME;
```

14. When D is in CONTEND state and receives an RRTS, it resets its defer_timer, sends RTS to the sender and sets its defer_timer enough to receive CTS from the sender. This behavior is not present in the control rules, either. But this is also implied in the discussion of RRTS packet in [Ref. 2].

```
defer_timer = TIME_SLOT + ROUND_TRIP_TIME;
```

3. Deferral Rules

QUIET state defer_timer values are based on the latest network topology knowledge of the node. When C overhears a packet from A to B, it does not make any assumption that it can also hear node B. So it sets its defer timer only to let A to receive the next packet from B.

1. When C hears an RTS packet from A to B, it goes to QUIET state from its current state and sets its defer_timer to let B to hear A's CTS packet.

```
defer_timer = TIME_SLOT;
```

2. When C hears a CTS packet from A to B , it goes to QUIET state from its current state and sets its defer_timer to let B to hear A's DS and DATA packets.

```
defer_timer = TIME_SLOT + (node_transmit_speed / sent_data_size);
```

3. When C hears a DS packet from A to B , it goes to QUIET state from its current state and sets its defer_timer sufficient for A to transmit DATA packet and hear B's ACK.

```
defer_timer = (node_transmit_speed / sent_data_size) + TIME_SLOT;
```

4. When B overhears an RRTS packet from C to D, it goes to QUIET state from its current state and sets its defer_timer to let RTS-CTS exchange happen between C and D.

```
defer_timer = 2 * TIME_SLOT;
```

4. Timeout Rules

1. When a station is in WFCONTEND state and its defer_timer expires, it sets its defer_timer to a random value which is determined by using remote backoff value of destination station, and goes to CONTEND_2 state.

```
defer_timer = TIME_SLOT * rand(BO_MIN...remote_backoff[destination_station]);
```

2. When a station is in CONTEND state and its defer_timer expires, it transmits an RTS to the destination, sets its timer enough to receive CTS and goes to WFCTS state.

```
defer_timer = TIME_SLOT + ROUND_TRIP_TIME
```

3. When a station is in CONTEND_2 state and its defer_timer expires, it sends RRTS to the destination and goes to IDLE state.

4. From any other state when the defer_timer expires, station goes to IDLE state.

III. SIMULATION

A. MOTIVATION FOR SIMULATION

Before building a system, analyzing its performance for the suitability to our needs is a good practice. Many add-on costs can be prevented that may surface when we start to use this system. Patching a built system is much more expensive then changing its specifications before building it.

Performance prediction of a protocol can only be done by thorough analysis of it. The analysis methods that can be used for this purpose are analytical and simulation methods. Analytical methods are based on mathematics. The mathematical representation of the system is built and relations among parameters are examined. During this procedure some assumptions are made about the system. These assumptions tend to diverge the model from real life. Simulation methods utilize computers. A software model of the system is generated and the behavior of the system is investigated. Many assumptions that are made for analytical representation of the system can be relaxed during the building of simulation model, so simulations can give more accurate results then analytical results.

B. SIMULATION PROGRAM (OPNET 2.4C)

To simulate the MACAW protocol, OPNET 2.4c from MIL 3, Inc. is used. OPNET is a sophisticated network design and simulation tool. It allows simulation from communications in a VLSI chip to Long Haul overseas networks. It uses an object-oriented approach to construct the desired architecture. There are three main building blocks. These are modules, nodes and networks. Networks are composed of nodes and/or other sub-networks. A node is analogous to a communicating device. It can be a workstation, a satellite, a telephone or an integrated circuit. Nodes are composed of modules. The behavior and operation of a node is determined by these modules. OPNET modules that are used to define a node are generators, queues, processors, transmitters, receivers, antennas and links among them.

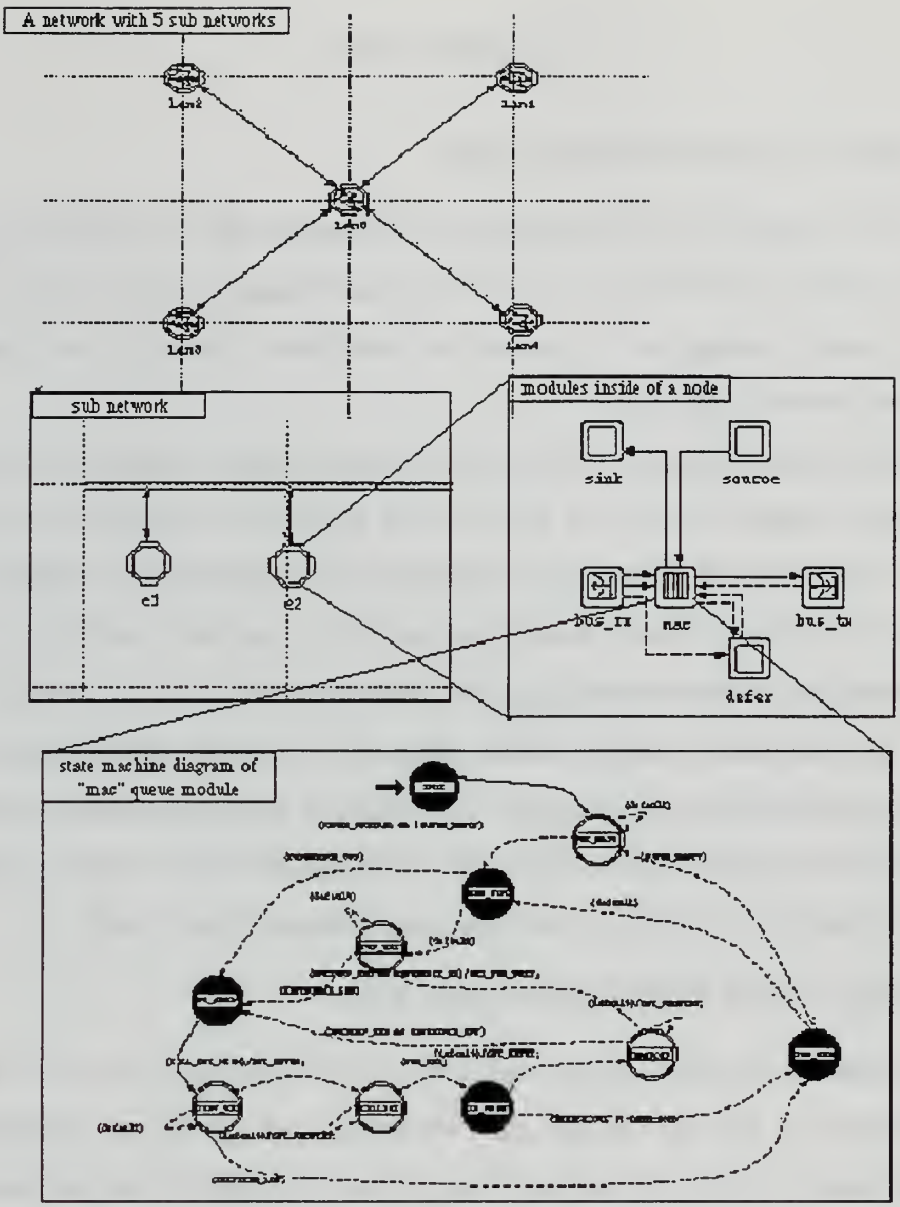


Figure 4. OPNET Building Blocks

Operations of the queue and the process modules are programmed as state machines. A state machine is composed of states and transitions. Actions are associated with the transitions and defined by using OPNET's C like programming language Proto-C and its libraries. The programmed module can either be active (executing some instructions) or idle (blocked) at time T. Operation of the state machine is interrupt driven. At time T, the process (programmed module) can only be either in one of states that it is composed of

or taking a transition from one state to another. If process is idle, it is in a state, if it is active it can be in a state or taking a transition from one state to another. A process receives interrupts while it is idle. When it receives an interrupt, it becomes active and begins to execute programmed actions.

A state in OPNET's state machine has two parts, entrance and exit, and these have some associated set of commands called executives in OPNET terminology. The entrance part can have some actions that should be performed when that state is entered (entrance executives). The exit part is similar, it can have some executable that should be executed before the state is left (exit executives). There are two types of states, these are called unforced and forced states in OPNET. These types determine how a state goes from its entrance section to its exit section. An unforced state blocks its execution when it finishes its entrance executives and waits for an interrupt. When it receives an interrupt it begins to execute its exit instructions and when these are done the transition that is true is taken to another state (a transition from a state back to itself is a valid transition). There must be one and only one transition true at this time. To have more than one true transitions or not to have any true transitions are error conditions. A forced state is analogous to a transient state. When this state is entered both entrance and exit instructions are executed one after the other and a valid outgoing transition is taken. There is no blocking in this type of state.

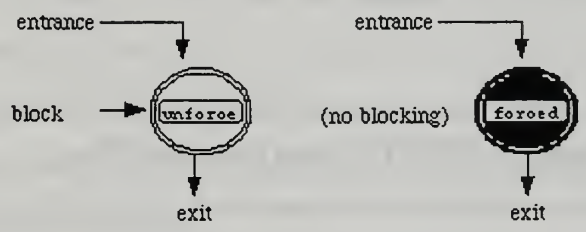


Figure 5. OPNET States

The execution order of actions in a programmed module is as follows. When an interrupt happens while a process (programmed module) is waiting (blocked) at the end of the entrance executives of an unforced state, the process goes to the exit part of that state and executes its exit instructions and when all of them are executed the valid outgoing

transition a process is taking, it is executed as well. When the next state is reached its entrance executives are executed. If this is an unforced state the execution is blocked. If this is a forced state, the process immediately goes to this new state's exit section and keep on performing actions in the same manner as above until it reaches an unforced state. When the process finally reaches an unforced state, it executes this unforced state's entrance executives and blocks its execution there until the next interrupt.

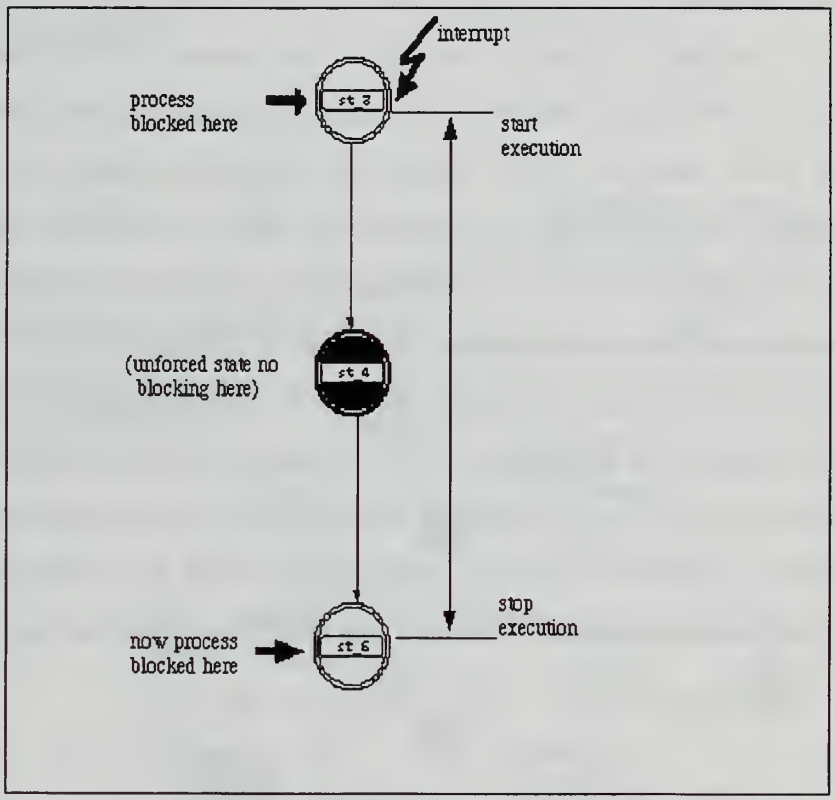


Figure 6. OPNET Process Execution Flow

Six modules are used to simulate a node that runs the MACAW protocol. These are

- Data generator,
- Passive queue,
- Protocol processor,
- Radio receiver,

- Radio transmitter and
- Antenna.

Data generator is a module without any input streams. It generates data packets with Poisson distribution for a given arrival rate. During all simulations data packet size is taken 520 bytes. When generator module generates the data packet module puts it into the passive queue.

Passive queue is an infinite length FIFO buffer. When a data packet enters this queue it remains there until the protocol processor module requests it. The delay time of a data packet in the node is calculated by taking the difference between its queue entrance time and queue leave time.

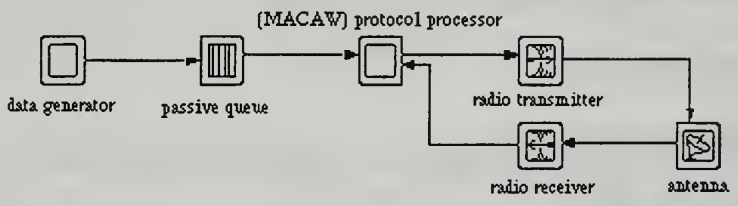


Figure 7. Module diagram of a Node Running MACAW Protocol

Protocol processor is the core unit of the simulation. This is the place where MACAW protocol runs. The protocol state machine is built in this process module. This state machine is built based on the state analysis done in [Ref. 5]. This study uses Systems of Communicating Machines [Ref. 6] to formally specify the protocol. Having a formally specified protocol eased the implementation process. States, transitions and transition conditions are taken as is, and only timer and backoff operations are added to complete the simulation model of the protocol.

When the protocol processor has data to send, it passes the packet to the radio transmitter module. This module puts the packet on the transmitting channel. Transmitter characteristics such as transmission rate, transmission frequency, modulation, bandwidth are defined in this module. Channel performs the communication between transmitter and receiver modules of other nodes in the network. OPNET refers to communication channels

as Transceiver Pipeline and simulates typical channel behaviors within this pipeline. When a packet enters into this communication pipeline, a series of actions are performed onto it one after the other to transport the packet to the receiver modules of other nodes. For a radio channel, 14 stages are pre-defined. Users can add more stages and/or change the actions in these stages. Default computation stages are (from sender to receiver):

- Receiver group establishment,
- Transmission delay,
- Link closure,
- Channel match,
- Transmitter antenna gain,
- Propagation delay,
- Receiver antenna gain,
- Received power,
- Background noise,
- Interference noise,
- Signal to noise ratio,
- Bit error rate,
- Error allocation and
- Error correction.

The first action, the receiver group establishment, is executed only once at the beginning of the simulation. This procedure is responsible for determining which nodes are authorized to receive the packets that this node broadcasts. This is done for all nodes in the network. The first five stages (after receiver group establishment) are executed at the

transmitter side and the rest is executed at the receiver side. A brief discussion of what other stages do is as follows.

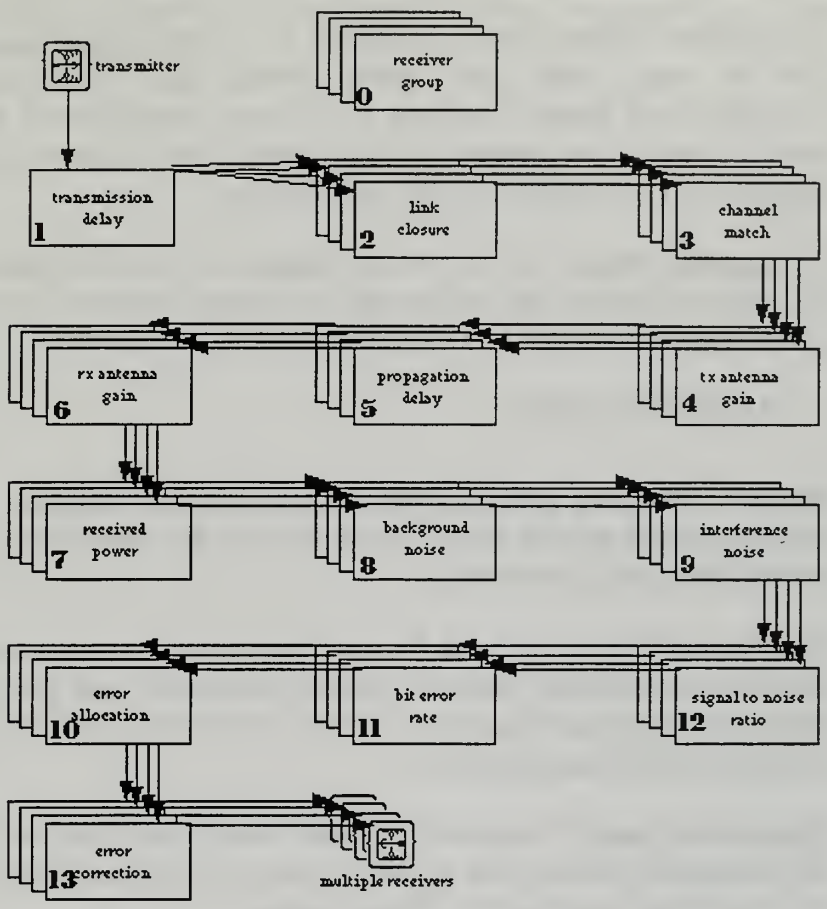


Figure 8. Radio Channel Transceiver Pipeline

1. At the Transmitter Side

- **Transmission delay:** Introduces the transmission delay to the packet according to its length and transmitter transmission speed. Since transmission delay only involves the transmitter parameters, it is calculated once per packet.

- **Link closure:** This procedure is responsible to determine whether the receiver is capable of receiving this packet. Calculations are based on the physical obstacles, like earth curvature, walls etc. This calculation is performed on the packet for all eligible receivers in the network.

- **Channel match:** Checks whether the transmitter's and receiver's modulation, frequency and bandwidth parameters match or not. This calculation is performed on the packet for all eligible receivers in the network.

- **Transmitter antenna gain:** Calculates the transmitter antenna gain. OPNET lets the users define their spatial antenna gain characteristics. For an omnidirectional antenna, the gain is 0. During simulation all transmitter and receiver antennas are assumed omnidirectional. This calculation is performed on the packet for all eligible receivers in the network.

- **Propagation delay:** Calculates the propagation delay of the packet according to the distance between the sender and the receiver module. This calculation is performed on the packet for all eligible receivers in the network.

2. **At the Receiver Side**

- **Receiver antenna gain:** Calculates the receiver's antenna gain. For an omnidirectional antenna, gain is 0. All receivers that received the packet on the channel perform this calculation.

- **Received power:** Computes the receiving power of the arrived packet (in watts) by considering distance between transmitter and the receiver, base frequency and transmitting power. All receivers that received the packet on the channel perform this calculation.

- **Background noise:** Calculates the thermal noise that effects the packet. During all simulations thermal noise at 290K is used as background noise. All receivers that receive the packet perform this calculation.

- **Interference noise:** Calculates the effects of interactions between transmissions. If a packet arrives at the same receiving channel while the receiver is already receiving another packet, this stage calculates their effect on each other. In MACAW simulation such a condition causes both packets become invalid.

- **Signal to noise ratio (SNR):** Calculates the signal to noise ratio of the received packet based on the information of background noise, interference and received power.

- **Bit error rate:** According to SNR information, bit error rate calculation is performed on the received packet.

- **Error allocation:** Computes the bits in error on the received packet, according to the information provided by the bit error rate calculation stage.

- **Error correction:** Determines the acceptability of the packet by the receiver module. The result depends on the error allocation stage outcomes and ability of receiver's error correction. In MACAW simulation, receivers have no error correction capability. So a packet that has bit errors is not forwarded to the receiver module.

IV. TEST CASES

A. PURPOSE

The purpose of the simulation is to determine the behavior of the MACAW protocol under different operational conditions. The behaviors that we are interested in are the utilization of the channel and the data packet mean queue delay of the transmitting nodes in the network. These characteristics are related to the communication channel load. The load of a network depends on the number of the nodes that are sharing the same transmission media and their data packet arrival rates. Besides these, the performance of the radio network that we are dealing with is heavily effected by the hidden-exposed node conditions.

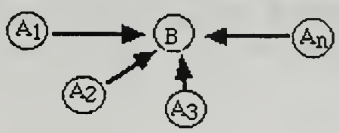
With these in mind, test cases are categorized into two main groups. The first group is to determine the effects of load on the protocol. The second group investigates hidden-exposed node behavior of the protocol.

B. LOAD CASES

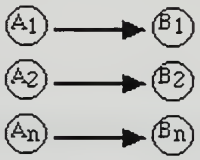
In the first group of cases, while investigating the load behavior of the protocol, it is also examined that whether the protocol acts in favor of any specific communication topology or not. For these purposes, four communication scenarios are defined for each topology. For each scenario, some test cases are generated with different numbers of nodes and packet arrival rates. These scenarios are as follows.

For all cases below, all nodes are in one cell, that is all can hear each other. Arrows show the direction of data streams.

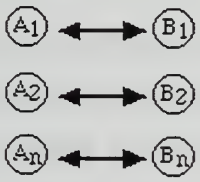
- **Reporting nodes:** All nodes in the network are transmitting to one particular receiving node.



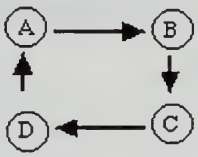
- **One way communicating nodes:** One node is transmitting to another receiving node.



- **Cross communicating nodes:** Nodes are communicating mutually.



- **Ring communication:** Each node in the network receives from a node and sends to another node, as shown.



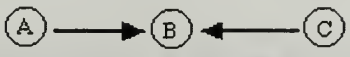
Each topology is tested up to four transmitting nodes from 10% to 100% total offered channel load. Offered channel load is adjusted by data arrival rates of the transmitting nodes. For example, to offer 30% load to the channel in a communication topology with two transmitting nodes, their total data packet arrival rate (R) is taken as (0.3 * channel_service_rate) and (R/2) data arrival rate is used in each transmitting node.

C. HIDDEN-EXPOSED NODE AND CELL CASES

The second category of cases investigates the behavior of the protocol for the nodes under exposed and/or hidden node and cell conditions. Test cases that are used for this purpose are as follows.

For the figures below, node B can hear node A and C but node A and C cannot hear each other.

- **Hidden node:** While A is sending to B, C wants to send to B too. In this scenario A and C are hidden from each other.



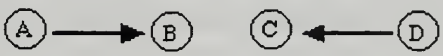
- **Exposed node:** While A is sending to B, B wants to send to C. Here B is exposed to A.

- **Hidden and exposed nodes:** While A and B are mutually exchanging data, C wants to send to B. Here A and C are hidden from each other and B is exposed to both A and B.

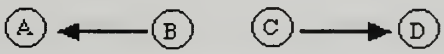
In addition to hidden and exposed node conditions, the effects of communications in neighboring cells are also investigated in this category. Communication scenarios regarding these topologies are as follows.

For the cases below node A and node B are in same cell, so does node C and node D. The nodes in the same cell can hear each other. In addition to that node B and node C can also hear each other.

- **Hidden Cell:** B and C are in different cells but they can hear each other. Node A is not aware of the communication going on in the adjacent cell since it cannot hear anything from that cell, the cell is hidden from A, the same is true for node D.

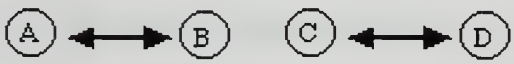


- **Exposed Cell:** B and C are in different cells but they can hear each others transmissions while they are communicating with other nodes in their cells (A and D respectively). So they are exposed to each others transmissions.



- **Exposed and Hidden Cells:** Node A does not hear any transmissions from the other cell while node C is hearing node B. So adjacent cell is hidden for A and exposed for C.

- **All in one:** There is a heavy communication in both cells.



V. TEST RESULTS

A. MEASURE FOR PERFORMANCE ANALYSIS

The most used performance measure for a protocol is network utilization. But this alone is not enough. Another important factor that should be considered is the data packet mean queue delay times in the transmitting nodes. Having a high utilization with very high delay times has no practical use, because users do not like to wait large amounts of times to complete a job. So we include data packet mean queue delay times in our performance measure criteria. The resultant formula is as follows.

$$Performance\_factor = \frac{Utilization\_of\_the\_network}{data\_mean\_queue\_delay} \quad (1)$$

According to this criteria, while comparing two networks (either running the same protocol or not), if the utilizations are equal, the one with lesser delay time will have a better performance measure.

B. DEFINITIONS

- **Bit_rate:** This is the transmission speed of the transceivers in the network in units of [bits/sec].
- **Service_rate:** This is the maximum number of data packets that a node can transmit/receive in a unit of time. Its unit is [number_of_data_packets / sec]. Service rate depends on many factors such as bit rate, data packet length, protocol overheads and process overheads. If we want to calculate the service_rate of a node analytically, we use the formula below.

$$service\_rate = \frac{1}{\frac{data\_pk\_length + protocol\_overheads}{bit\_rate} + process\_delays} \quad (2)$$

Data packet queue delay times are relative to service_rate of the node. While 1 sec delay time can be very high for a network with a transmission speed of 1Mbps and data length of 1Kb (1000 data packets -1Mb- can be queued during this time), it is normal for a

network that has a transmission speed of 1Kbps and data length of 1Kb (only 1 data packet - 1Kb- can be queued/dequeued during this period). We can make our performance measure free from this effect by simply normalizing the delay times with respect to the service rate of the network.

$$normalized\_delay\_time = mean\_delay\_time\ [\ \text{sec}\ ] \times service\_rate\ [data\_pk\ /\ \text{sec}\ ] \quad \textbf{(3)}$$

This formula gives us a delay time in terms of the number of data packets. During our simulations we used fixed bit_rate and data packet length so we do not need to normalize delay times for our comparisons. But we will also give the normalized results of the performance measures for future comparisons/evaluations.

- **Offered_channel_load:** This is the estimated number of data packets that will be put into the channel during one unit of time. This is adjusted by data packet arrival rates of the transmitting nodes. If we want to load the channel 30% with 3 transmitting nodes, we choose an arrival rate of (service_rate × 0.3) / 3 for each transmitting node in the network.

- **BO_MIN:** Minimum backoff constant. Backoff value cannot be smaller then this value. This is set prior to construction of the network, by the users. During simulations this is set to 3.

- **BO_MAX:** Maximum backoff constant. Backoff cannot be greater then this value. This is also another parameter that has to be set prior to network construction. This is set to 64 for all simulations.

C. SIMULATION VALUES

Constant values that are used during the simulations are listed below. The values of backoff limits and packet lengths are taken to be consistent with the original paper [Ref. 2]. The length of the data packet that is generated by the generator module is 512 bytes, but 8 bytes of header information (Destination Address) is added to this data packet before passing it to the lower layer by the generator module. So MACAW protocol processor sees the data packet as 520 bytes (4160 bits). The values of simulation period and bit rate are taken to have a period long enough for the protocol to complete its transient state and not to get bored while waiting for simulations to finish. To have a large bit rate and simulation

period means to have more events to simulate (since the network will produce more packets per unit of time) and to wait more to complete the simulation.

- *simulation_period* = 400 sec
- BO_MIN = 3
- BO_MAX = 64
- *bit_rate* = 40960 bits / sec
- *data_packet_length* = 4160 bits
- *control_packet_length* = 240 bits

To successfully send a data packet to destination, a minimum of 4 control packets are needed (RTS, CTS, DS, ACK). Also a header of 30 bytes is added to each data packet. So our service rate is

$$service\_rate = {}^{1}\!\Big/\!\left(((4160 + 240) + 4*240)/4960\right) = 7.6\ [data\_pk/\text{sec}\,] \tag{4}$$

D. BASE CASE

The first case in the simulations is to determine the maximum performance of the MACAW protocol for a given data packet arrival rate. For this purpose a simple sender-receiver network is constructed in the OPNET network simulation environment.

To offer 10% to 100% load to the channel between node A and node B, arrival rates of 1 to 8 [data_pk/sec] are used. Since we have calculated the maximum service rate of the channel (7.6[pk/sec]), the offered load to the channel can easily be calculated as follows

$$Load = {}^{arrival\_rate}\!\Big/_{channel\_service\_rate} \tag{5}$$

Data mean queue delay times and utilization of the transmitter are obtained by using OPNET's analysis tools. Related information is collected during the simulation. The formula that is used to calculate the utilization of the network at time t is:

$$Utilization(t) = total\_data\_transmission\_time / t \quad (6)$$

Total data transmission time (*total_data_transmission_time*) is obtained by counting the successful data transmissions only (i.e. unsuccessful transmissions are not counted). Queue delay time of a data packet is measured by OPNET. It keeps track of data queue entry and exit times. This is done for all packets arrived to the queue during one unit of time (1 sec) and their mean value is calculated as "mean queue delay time" of the data packet at time t. Their average value is taken as network's utilization and data mean queue delay times. If the interested parameter does not complete its transient state (i.e. not become stable) then its last value (at time 400 sec) is read. This is the case for increasing mean delay times under heavy offered channel loads.

Arrival Rate [data_pk/sec]	1	2	3	4	5	6	7	8
Channel Offered Load	13.33%	26.67%	40.00%	53.33%	66.67%	80.00%	93.33%	106.67%
Data Queue Mean Delay [sec]	0.13	0.31	0.5	1.3	2.3	3.5	18	35
Utilization	0.11	0.21	0.31	0.42	0.5	0.58	0.66	0.66
Performance (non-normalized)	0.85	0.68	0.62	0.32	0.22	0.17	0.04	0.02
Performance (normalized) x 100	11.13	8.91	8.16	4.25	2.86	2.18	0.48	0.25

Table 3. Base Case Simulation Results

Performance and normalized performance are calculated as discussed in the "Measure For Performance Analysis" section of this chapter.

$$Performance\_measure = \frac{Utilization\_of\_the\_network}{data\_mean\_queue\_delay} \quad (7)$$

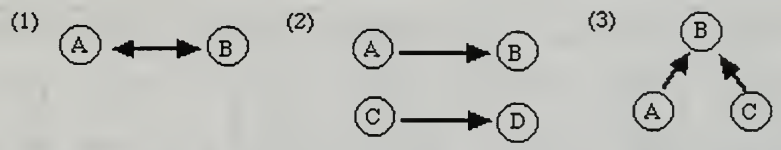
$$normalized - Performance = \frac{Utilization\_of\_the\_network}{data\_mean\_queue\_delay \times service\_rate} \quad (8)$$

In this test case, there is no interference to the transmitting node, so it should work at top performance for given arrival rates. These results provides us a base to compare our simulation results with each other. As expected, the performance decreased rapidly above 80% of offered load due to the high data queue mean delay times. Maximum utilization for the protocol with given packet lengths, backoff limits and data packet arrival distribution method is observed as 66%.

E. LOAD CASE RESULTS

In the next section we will give the results of the first category test cases that are grouped by the number of transmitting nodes.

1. Cases with Two Transmitting Nodes



Three topologies are used, (1) one pair mutually communicating nodes (cross communicating nodes) (2) two one-way transmitting nodes and their corresponding receiver nodes and (3) two reporting nodes.

For low offered channel loads (up to 50%), all three of them have pretty similar performance measures (Table 4) . But as we increase the offered load on the channel (by increasing arrival rates in the transmitting nodes) each showed unique behavior. This can be easily seen in Figure 9. Among them, topology (1) gave the best results with smaller mean delay times and higher utilization, compared to the other two. The reason for this behavior is, the backoff values in network (1) are lower during the simulation period than in (2) and (3).

The scenario that causes this is as follows. At time T1 Node A has a local backoff of 3 and remote backoff of 9 for node B (so B has a local backoff of 9 and remote backoff of 3 for A, since A and B are exchanging data packets). At time T2 both transmits RTS packets and they collide so both increase their remote backoff values for each other. Thus,

A's remote backoff for B becomes 13 and B's backoff value for A becomes 4. It is more likely that B will transmit first because it has a lower backoff value. When B transmits an RTS to A, it will set local backoff field of the packet to 9 and remote backoff field to 4. When A receives this packet it will copy the value in the local backoff field as B's remote backoff. So remote backoff of B in A goes from 13 to 9. A quick increase in B's remote backoff value in A. Backoff increases that happened after a series of collisions will be pulled down to its pre-collision value after a successful transmission. Thus, the backoff values in the nodes remains low and they try to acquire channel more rapidly. This certainly causes more collisions of packets but persistence of nodes helps them to achieve high performance because of their lower mean delay times.

Channel Offered Load	13.16%	26.32%	39.47%	52.63%	78.95%	92.11%	105.26%
Network(1)							
Average Data Queue Mean Delay [sec]	0.09	0.182	0.375	0.78	40.5	32	56.75
Worst Case Delay [sec]	0.09	0.2	0.42	0.92	50	32	100
Channel Utilization	0.108	0.2	0.3	0.4	0.5	0.56	0.6
Performance (Non-normalized)	1.200	1.099	0.800	0.513	0.012	0.018	0.011
Performance (Normalized)×100	15.79	14.46	10.53	6.75	0.16	0.23	0.14
Network(2)							
Average Data Queue Mean Delay [sec]	0.088	0.155	0.525	2.95	28	59	75.7
Worst Case Delay [sec]	0.088	0.16	0.65	5.4	28	117	150
Channel Utilization	0.108	0.21	0.31	0.41	0.54	0.5	0.53
Performance (Non-normalized)	1.227	1.355	0.590	0.139	0.019	0.008	0.007
Performance (Normalized)×100	16.15	17.83	7.77	1.83	0.25	0.11	0.09
Network(3)							
Average Data Queue Mean Delay [sec]	0.085	0.245	1.245	11	72.5	91	108
Worst Case Delay [sec]	0.085	0.25	1.53	12	75	100	110
Channel Utilization	0.108	0.21	0.3	0.4	0.4	0.4	0.4
Performance (Non-normalized)	1.271	0.857	0.241	0.036	0.006	0.004	0.004
Performance (Normalized)×100	16.72	11.28	3.17	0.48	0.07	0.06	0.05

Table 4. Two Transmitting Nodes Simulation Results

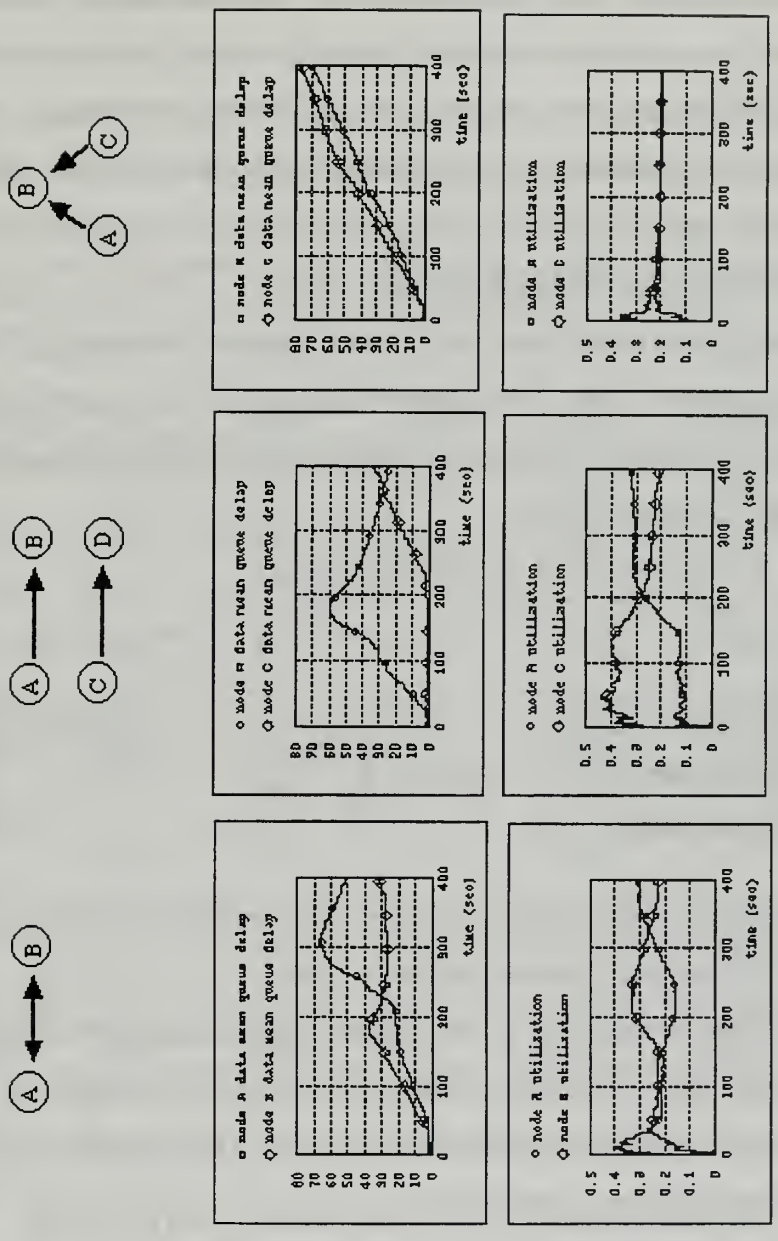
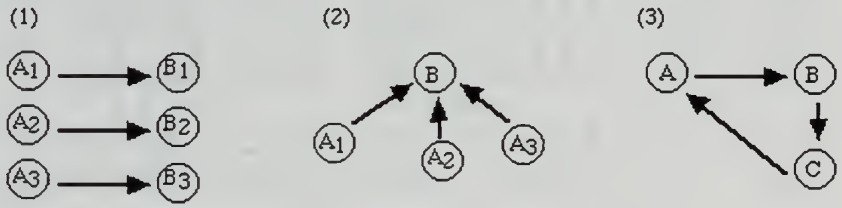


Figure 9. Two Transmitting Nodes Mean Delay and Utilization Graphs for 80% Offered Load

On the other hand in (2) transmitters can have quite different backoff values from each other. For high loads, the node that picks smaller backoff values to transmit its receiver one after the other will eventually have smaller remote backoff value for that receiver than the others (each successful transmission decreases the remote backoff value in the sender). So this node will have a greater chance to acquire the channel more than the others. Having small remote backoff means to wait less an average to acquire the channel. So while others are waiting to get the channel with high backoff values, this "quicker node" will get the channel again and again because of its high arrival rate and low remote backoff value. This prevents fair access of the nodes to the channel.

In case (3), both transmitting nodes shares the same remote backoff value since both transmitting to the same node. This guarantees fair access to the channel but rapid increase of backoff value for receiving node causes high data queue delays in transmitters, so we get low performance from the network.

2. Cases with Three Transmitting Nodes



Three topologies are used (1) three one-way transmitting nodes, (2) three reporting nodes and (3) ring like data exchange with three nodes.

For low and medium (up to 50%) offered channel loads, all three networks show similar behaviors. Above 50% offered channel load, the backoff algorithm determines the performance of the network. Among them topology (1) has better total performance measure compared to the other two networks.

One-way communicating network (1) of this case has also similar behavior as in two transmitting nodes case. The quicker node gets the channel and holds it for high offered loads of the network. This causes other nodes to have much greater mean delay times so

lesser performance measures. The nature of the backoff algorithm prevents the non communicating nodes' backoff values be effected from each other. Each node copies the local backoff values of the other nodes that it overhears into a separate location of its backoff tables and uses these values when it wants to communicate one of these nodes. These values in the backoff tables are mutually exclusive, they do not effect each other at all. While one node has very small backoff values, other node in the cell may have very large values. This is also what is happening in the one-way communicating nodes cases.

Channel Offered Load	11.84%	19.74%	39.47%	59.21%	78.95%	98.68%
Network(1)						
Average Data Queue Mean Delay [sec]	0.045	0.093	0.330	7.817	30.000	116.667
Worst Case Delay [sec]	0.052	0.100	0.430	14.000	50.000	200.000
Channel Utilization	0.098	0.155	0.3	0.45	0.47	0.436
Performance (Non-normalized)	2.162	1.673	0.909	0.058	0.016	0.004
Performance (Normalized)×100	28.44	22.01	11.96	0.76	0.21	0.05
Network(2)						
Average Data Queue Mean Delay [sec]	0.065	0.123	0.610	15.667	71.667	91.667
Worst Case Delay [sec]	0.067	0140	0.700	20.000	85.000	100.000
Channel Utilization	0.096	0.156	0.3	0.426	0.43	0.435
Performance (Non-normalized)	1.485	1.265	0.492	0.027	0.006	0.005
Performance (Normalized)×100	19.53	16.64	6.47	0.36	0.08	0.06
Network(3)						
Average Data Queue Mean Delay [sec]	0.045	0.120	0.360	14.100	58.000	70.000
Worst Case Delay [sec]	0.045	0.150	0.400	20.000	67.000	80.000
Channel Utilization	0.099	0.156	0.3	0.45	0.45	0.49
Performance (Non-normalized)	2.200	1.300	0.833	0.032	0.008	0.007
Performance (Normalized)×100	28.95	17.11	10.96	0.42	0.10	0.09

Table 5. Three Transmitting Nodes Simulation Results

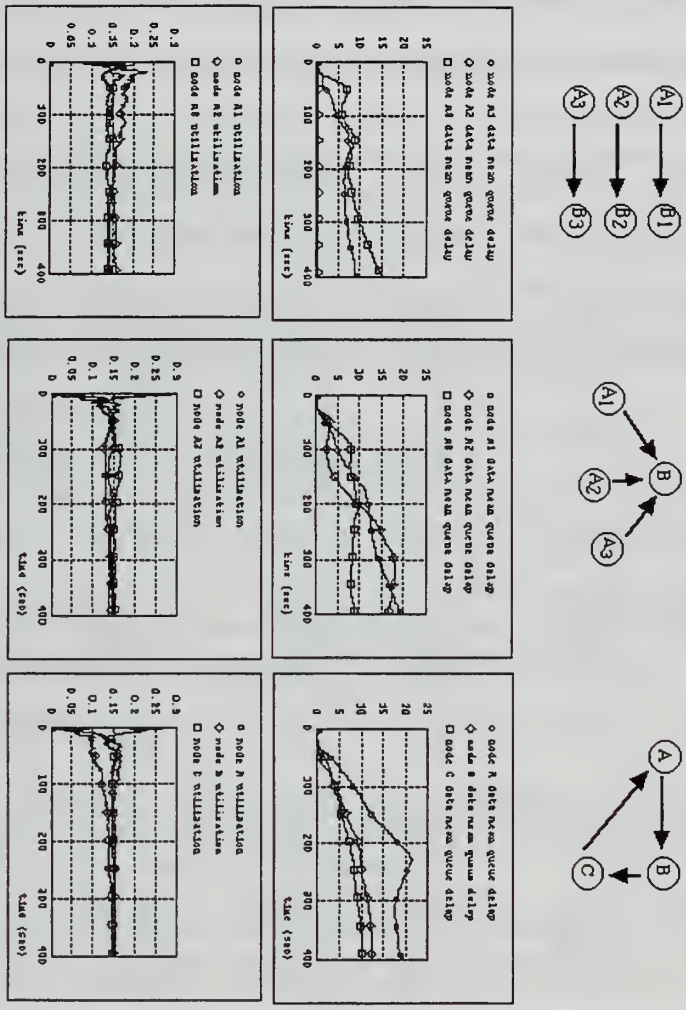
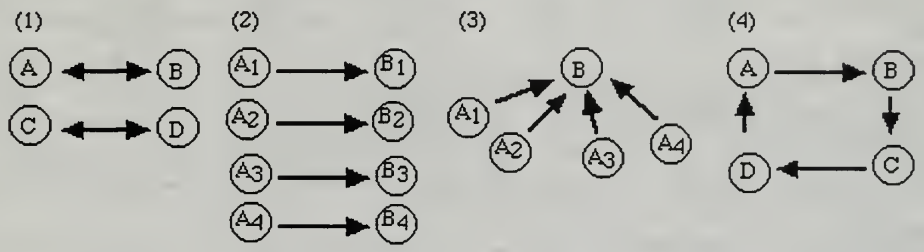


Figure 10. Three Transmitting Nodes Mean Delay and Utilization Graphs for 52% Load

Reporting nodes (2) behavior is the same as in two transmitting nodes case. Channel access is fair but performance is low, because of high backoff values during the communication.

Ring (3) communication results is the same as reporting nodes case but slightly better. Because there is no receiver bottleneck as in reporting nodes.

3. Cases with Four Transmitting Nodes



Four networks are used, (1) two pairs of mutually communicating nodes, (2) four one-way communicating nodes, (3) four reporting nodes and (4) circular data exchange with four nodes.

Performance measures of all networks are similar up to 50% offered channel loads. After that point, the backoff algorithm determines the behavior of the network. The results of mutually communicating network (1) is similar to the results of mutually communicating network of "two transmitting nodes" case. When RTS packets are transmitted to each other from node A and node B collide, the remote backoff values in these nodes for each other are increased (because they cannot receive CTS for their RTS). But in the first successful data exchange this increase is pulled down to its pre-collision value. This is very effective to keep backoff values low when no other node is transmitting in the cell as in two transmitting nodes case, however in this topology there are two more nodes that node A's transmissions can collide. So the effect of this behavior is weakened.

Case (2) shows the expected behavior. In high loads (above 50%) the quicker node gets the channel with lower backoff value for its receiver and keeps it .This causes other nodes to have much greater backoff values, thus to have lower performance measures. This behavior is the same as two and three transmitting nodes cases.

Channel Offered Load	10.53%	15.79%	26.32%	52.63%	78.95%	105.26%
Network(1)						
Average Data Queue Mean Delay [sec]	0.052	0.064	0.125	1.415	47.500	85.750
Worst Case Delay [sec]	0.057	0.080	0.153	2.230	60.000	104.000
Channel Utilization	0.081	0.12	0.218	0.403	0.45	0.45
Performance (Non-normalized)	1.565	1.868	1.744	0.285	0.009	0.005
Performance (Normalized)×100	20.59	24.58	22.985	3.75	0.12	0.07
Network(2)						
Average Data Queue Mean Delay [sec]	0.122	0.092	0.115	1.335	46.312	131.350
Worst Case Delay [sec]	0.130	0.100	0.124	1.900	65.000	300.000
Channel Utilization	0.083	0.124	0.205	0.394	0.45	0.48
Performance (Non-normalized)	0.678	1.341	1.775	0.295	0.010	0.004
Performance (Normalized)×100	8.92	17.64	23.35	3.88	0.13	0.05
Network(3)						
Average Data Queue Mean Delay [sec]	0.050	0.062	0.127	1.823	52.500	88.250
Worst Case Delay [sec]	0.064	0.076	0.148	2.500	60.000	91.000
Channel Utilization	0.079	0.126	0.212	0.392	0.341	0.45
Performance (Non-normalized)	1.588	2.024	1.669	0.215	0.006	0.005
Performance (Normalized)×100	20.89	26.63	21.96	2.83	0.09	0.07
Network(4)						
Average Data Queue Mean Delay [sec]		0.046	0.107	2.075	49.000	84.500
Worst Case Delay [sec]		0.054	0.125	3.000	70.000	88.000
Channel Utilization		0.118	0.213	0.4	0.46	0.44
Performance (Non-normalized)		2.551	1.986	0.193	0.009	0.005
Performance (Normalized)×100		33.57	26.13	2.54	0.12	0.07

Table 6. Four Transmitting Nodes Simulation Results

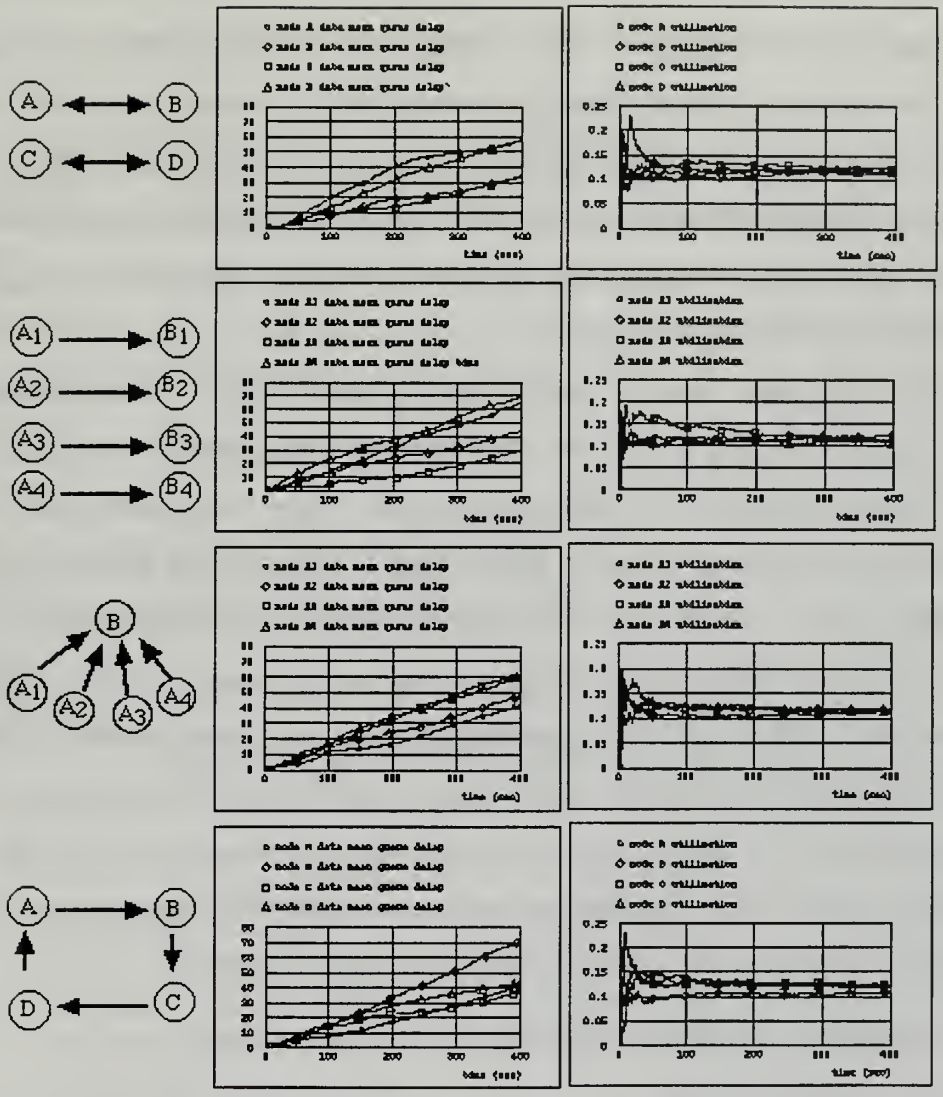


Figure 11. Four Transmitting Nodes Mean Delay and Utilization Graphs for 80% Load

Case (3) also gives similar results as in two and three transmitting nodes, reporting nodes cases. Channel access of the transmitting nodes are pretty fair but they have high remote backoff values for the receiver which causes high data mean queue delays and low performance.

Circular data exchange (4) network does not have any particular different behavior than reporting nodes case.

4. Summary

Channel utilization almost never exceeds 50% for all simulations. The exceptions are the base case and "one pair mutually communicating" network case. In the base case there is no interfering node against the transmitting node, so we should get "the most" from the protocol. The reason that one pair mutually communicating network has high utilization and low backoff values than the other simulation cases is it has a slower rate of backoff increase for its remote backoff values.

For high loads, one-way communicating nodes always violate the fairness of channel access. Reporting (3) and ring (4) cases have similar results. They both have fair channel access in all times with high backoff values so lower total performance measures than the other network topologies in their category. To have high backoff value causes larger delay times on average therefore low performance measures in these networks.

The best performance measure observed during the simulations are at 20% channel load conditions of three and four transmitting nodes cases. As the number of transmitting nodes in the network increases, the effect of having high backoff values decreases. Because the chance that there is a node ready to transmit when the channel is free is high. So idle times of the channel is used and total performance measure of the network increases, since per node utilization and data mean queue delay times does not change.

F. HIDDEN-EXPOSED NODE AND CELL CASE RESULTS

1. Exposed Node

In this case, A and C cannot hear each other but B can hear both. In this scenario B is exposed to A. Up to 80% offered load on node B, performance measure of both A and B is good (close to "one pair mutually communicating node case" of load cases). Above this threshold the "lucky" node gets the initiative and transmits more often than the other (as in the one-way transmitting nodes case of load cases).

Arrival Rate Per Node [data_pk/sec]	0.5		1		1.5		2	
Common Channel Offered Load	13.16%		26.32%		39.47%		52.63%	
Node	A	B	A	B	A	B	A	B
Data Queue Mean Delay	0.1	0.07	0.18	0.16	0.46	0.5	0.6	0.5
Node Utilization	0.057	0.052	0.1	0.1	0.15	0.16	0.2	0.2
Performance (non-normalized)	0.570	0.743	0.556	0.625	0.326	0.320	0.333	0.400
Performance (normalized) x 100	7.50	9.77	7.31	8.22	4.29	4.21	4.39	5.26

Arrival Rate Per Node [data_pk/sec]	3		4	
Common Channel Offered Load	78.95%		105.26%	
Node	A	B	A	B
Data Queue Mean Delay	22	41	118	125
Node Utilization	0.26	0.26	0.4	0.14
Performance (non-normalized)	0.012	0.006	0.003	0.001
Performance (normalized) x 100	0.16	0.08	0.04	0.01

Table 7. Exposed Node Case Simulation Results

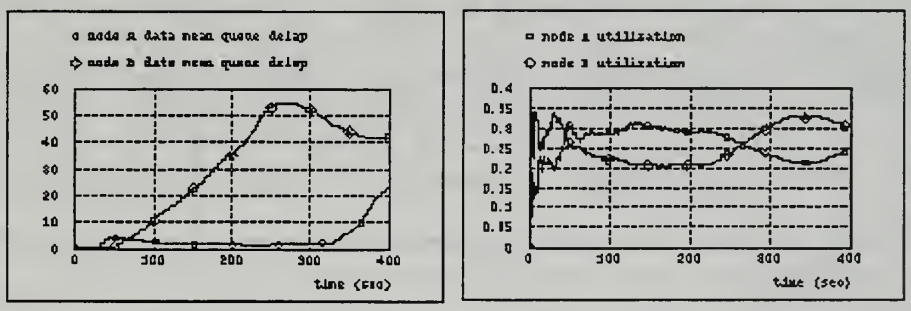


Figure 12. Exposed Node Mean delay and Utilization Graph for 80% Common Channel Load

When the RTS packets of A and B collide, they increase their remote backoff values of the receiving node in their backoff tables, since they cannot receive a CTS response for their RTS packet (A increases the remote backoff of B, B increases the remote backoff of C). Eventually one of them (the "lucky" one) transmits successfully first and decreases its remote backoff value for the receiver. If this scenario happens a couple of times in favor of

the "lucky" node, the remote backoff values of transmitting nodes become great. Every similar event provides more chance of channel acquirement to the lucky node. Finally it gets the channel and holds it while other is deferring to transmit. This scenario is pretty much the same as one-way communicating nodes cases of load cases.

2. Hidden Node

In this case, A and C cannot hear each other but B can hear both, so A and C are hidden from each other. In this scenario, channel access (in this case, channel is node B) of nodes A and C is always fair but in moderate and high loads (above 40%) on node B, total performance measure is lower than the exposed node case due to the high backoff value of the receiver (node B). The rapid increase of backoff value makes node A and C pick larger delay times in average, and defer accordingly. Result is high mean delay times in the transmitting nodes. This behavior is the same as in two reporting nodes case of load cases.

Arrival Rate Per Node [data_pk/sec]	0.5		1		1.5		2	
Common Channel Offered Load	13.16%		26.32%		39.47%		52.63%	
Node	A	C	A	C	A	C	A	C
Data Queue Mean Delay	0.07	0.12	0.25	0.25	1.13	0.84	15	13
Node Utilization	0.05	0.057	0.1	0.1	0.16	0.14	0.197	0.197
Performance (non-normalized)	0.714	0.475	0.400	0.400	0.142	0.167	0.013	0.015
Performance (normalized) x 100	9.40	6.25	5.26	5.26	1.86	2.19	0.17	0.20

Arrival Rate Per Node [data_pk/sec]	3		4	
Common Channel Offered Load	7.95%		105.26%	
Node	A	C	A	C
Data Queue Mean Delay	80	80	120	103
Node Utilization	0.194	0.194	0.18	0.2
Performance (non-normalized)	0.002	0.002	0.002	0.002
Performance (normalized) x 100	0.03	0.03	0.02	0.03

Table 8. Hidden Node Case Simulation Results

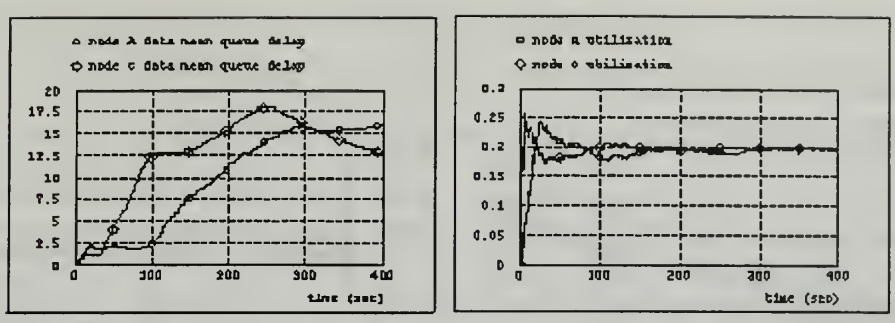


Figure 13. Hidden Node Mean Delay and Utilization Graph for 52% Common Channel Load

3. Hidden and Exposed Nodes

In this case, A and C cannot hear each other but B can hear both. Here, A and C are hidden from each other while B is exposed to both.

Arrival Rate Per Node [data_pk/sec]	0.2			0.3			0.5		
Common Channel Offered Load	7.89%			11.84%			19.74%		
Node	A	B	C	A	B	C	A	B	C
Data Queue Mean Delay	0.05	0.056	0.056	0.045	0.07	0.068	0.13	0.074	0.094
Node Utilization	0.02	0.02	0.02	0.029	0.029	0.031	0.053	0.053	0.053
Performance (non-normalized)	0.400	0.357	0.357	0.644	0.414	0.456	0.448	0.716	0.564
Performance (normalized) x 100	5.26	4.70	4.70	8.48	5.45	6.00	5.36	9.42	7.42

Arrival Rate Per Node [data_pk/sec]	1			1.5			2		
Common Channel Offered Load	39.47%			59.21%			78.95%		
Node	A	B	C	A	B	C	A	B	C
Data Queue Mean Delay	0.3	0.3	0.33	24	11	18	74	51	56
Node Utilization	0.096	0.1	0.1	0.14	0.15	0.14	0.14	0.14	0.14
Performance (non-normalized)	0.320	0.333	0.303	0.006	0.014	0.008	0.002	0.003	0.003
Performance (normalized) x 100	4.21	4.39	3.99	0.08	0.18	0.10	0.02	0.04	0.03

Table 9. Hidden-Exposed Node Case Simulation Results

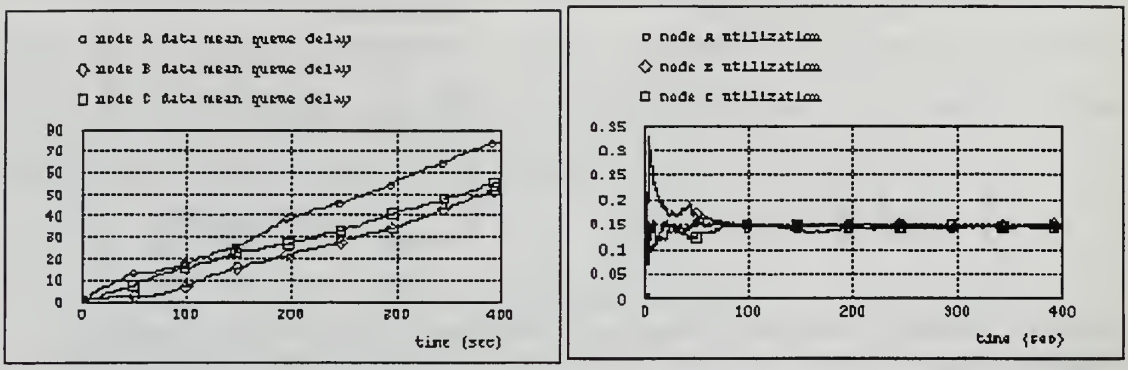


Figure 14. Hidden-Exposed Node Mean Delay and Utilization Graphs for 52% of Common Channel Load

The behavior of this network is closer to the hidden node case. After 40% of load over node B, data mean queue delay times of the transmitting nodes increase dramatically. Different from hidden and exposed node cases, the highest total performance measure for this network is observed at 20% load condition, as in three and four transmitting nodes of load cases.

4. Exposed Cell

In this topology, there are two cells composed of nodes {A, B} and {C, D} respectively (curly braces denotes a cell with nodes as its elements). Node B and node C can hear each other, therefore B is exposed to the communications in the cell {C, D} so does node C is exposed to the communications in the cell {A,B}.

Channel access of node B and C is fair. For high loads (above 75%) on the channel between B and C, mean delay times are high as expected. This is due to the quiet periods of these nodes. Quite period is the time that the node waits for other to complete its transmissions (result of being in QUIET state). In this network, there is no congestion at the receiver side, so the backoff values of the receivers are low. For high loads, when a node gets the channel, it transmits all its pending data packets one after the other, since backoff value of the receiver is low. Eventually it is out of data packets and releases the channel, then the other transmitter node gets the channel and transmits its pending data packets. This

can be seen in the mean delay time graph of the nodes (Figure 15). While the node that keeps the channel has a constant or decreasing mean delay time, the other node has an increasing mean delay time. When deferring node gets the channel, its delay times becomes constant or decrease while other's increases.

Arrival Rate Per Node [data_pk/sec]	0.3		0.5		1		2	
Common Channel Offered Load	7.89%		13.16%		26.32%		52.63%	
Node	A	C	A	C	A	C	A	C
Data Queue Mean Delay	0.049	0.049	0.08	0.072	0.13	0.175	0.65	0.45
Node Utilization	0.031	0.034	0.055	0.05	0.1	0.11	0.2	0.21
Performance (non-normalized)	0.633	0.694	0.688	0.694	0.769	0.629	0.308	0.467
Performance (normalized) x 100	8.32	9.13	9.05	9.14	10.12	8.27	4.05	6.14

Arrival Rate Per Node [data_pk/sec]	3		4	
Common Channel Offered Load	78.95%		105.26%	
Node	A	C	A	C
Data Queue Mean Delay	12.2	8	70	10
Node Utilization	0.3	0.3	0.15	0.44
Performance (non-normalized)	0.025	0.037	0.002	0.044
Performance (normalized) x 100	0.32	0.49	0.03	0.58

Table 10. Exposed Cell Case Simulation Results

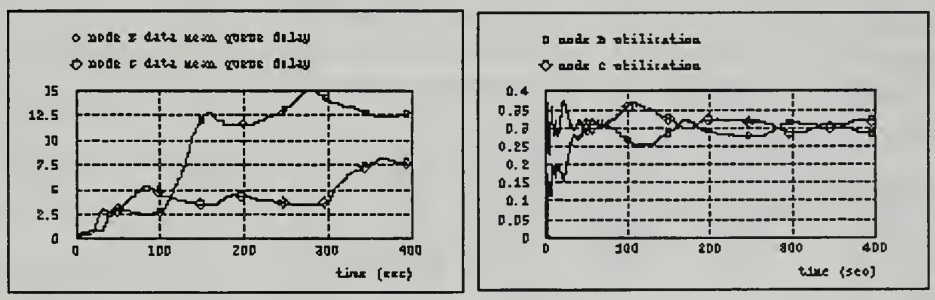
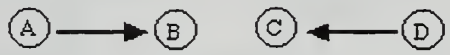


Figure 15. Exposed Cell Mean Delay and Utilization Graph for 80% Common Channel Load

In this network the total utilization is high (60% for 80% common channel load) due to the small backoff values of the receivers. The similar effect was observed "one pair communicating network" of load cases.

5. Hidden Cell



{A, B} is in a cell while {C, D} is in another. A is not aware of the cell {C, D} so this cell is hidden from A. The same is also true for D. Cell {A,B} is hidden from node D. Nodes B and C can hear each other even though they are in different cells.

Channel access of A and D (transmitter nodes) is fair but due to the congestion at the receivers, backoff values of the receivers are high so does data mean queue delay times. This is significant after the cell loads of 30%.

Arrival Rate Per Node [data_pk/sec]	0.3		0.5		1		2	
Cell Channel Offered Load	3.95%		6.58%		13.16%		26.32%	
Node	A	D	A	D	A	D	A	D
Data Queue Mean Delay	0.085	0.1	0.12	0.18	0.4	0.25	1.3	1.15
Node Utilization	0.029	0.033	0.05	0.054	0.11	0.1	0.19	0.2
Performance (non-normalized)	0.341	0.330	0.417	0.300	0.275	0.400	0.146	0.174
Performance (normalized) x 100	4.49	4.34	5.48	3.95	3.62	5.26	1.92	2.29

Arrival Rate Per Node [data_pk/sec]	3		4	
Common Channel Offered Load	39.47%		52.63%	
Node	A	D	A	D
Data Queue Mean Delay	40	37	90	42
Node Utilization	0.25	0.27	0.24	0.3
Performance (non-normalized)	0.006	0.007	0.003	0.007
Performance (normalized) x 100	0.08	0.10	0.04	0.09

Table 11. Hidden Cell Case Simulation Results

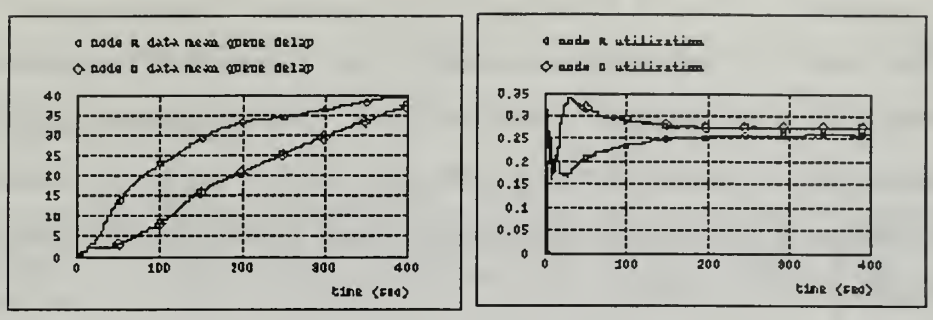
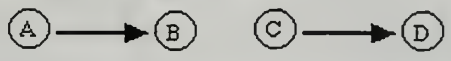


Figure 16. Hidden Cell Mean Delay and Utilization Graph for 40% Cell Load

6. Hidden and Exposed Cell



{A,B} and {C,D} are in different cells but node B an C can hear each other. In this network, cell {C,D} is hidden from node A and exposed to node B. Also node C is exposed to cell {A,B}.

Arrival Rate Per Node [data_pk/sec]	0.3		0.5		1		2	
Cell Channel Offered Load	3.95%		6.58%		13.16%		26.32%	
Node	A	C	A	C	A	C	A	C
Data Queue Mean Delay	0.09	0.045	0.1	0.073	0.42	0.11	8	0.34
Node Utilization	0.032	0.031	0.055	0.055	0.11	0.1	0.21	0.21
Performance (non-normalized)	0.356	0.689	0.550	0.753	0.262	0.909	0.026	0.618
Performance (normalized) x 100	4.68	9.06	7.24	9.91	3.45	11.96	0.35	8.13

Arrival Rate Per Node [data_pk/sec]	3		4	
Common Channel Offered Load	39.47%		52.63%	
Node	A	C	A	C
Data Queue Mean Delay	100	0.66	130	0.8
Node Utilization	0.16	0.34	0.14	0.4
Performance (non-normalized)	0.002	0.515	0.001	0.500
Performance (normalized) x 100	0.02	6.78	0.01	6.58

Table 12. Hidden-Exposed Cell Case Simulation Results

As expected node C is the winner in this topology for above moderate cell loads (30%). Since there is no congestion at the receiver of node C, node D will have smaller backoff values comparing to the receiver of A which is node B. Thus, the average backoff of node C will be lower then node A, and C will have the opportunity to acquire the common channel (channel between B and C) more often then A. This is quite significant for high common channel loads.

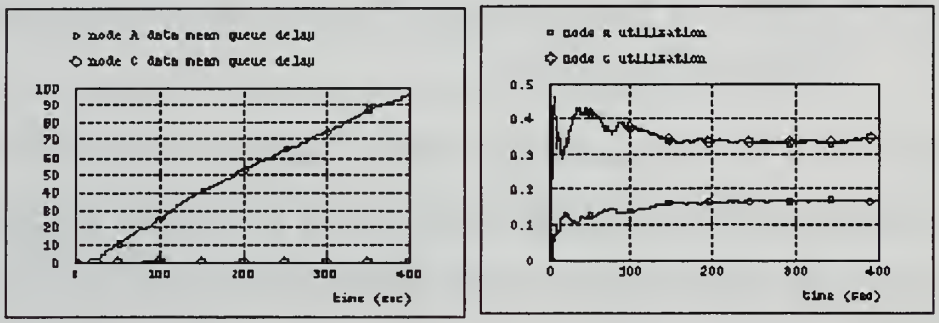
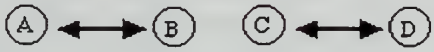


Figure 17. Hidden-Exposed Cell Mean Delay and Utilization Graph for 40% of Cell Load

7. All in One



In this case, there is heavy communication in both cell {A,B} and cell {C,D}. Nodes B and C can hear each other.

Up to 25% of cell loads, channel access is fair for all transmitting nodes. But above this threshold one of "inner nodes" (node B or C) grabs the common channel (channel between nodes B and C) and causes other nodes in the network defer while it transmits its data packets one after the other. This causes very high delay times for other nodes. Backoff values of the inner nodes are high compared to outer nodes (nodes A and D), because of the congestion levels at the place of the nodes. Since the backoff values are low for A and D, node B and C will have the chance of getting and holding the channel for high loads.

Arrival Rate Per Node [data_pk/sec]	0.2				0.3			
Per Cell Channel Offered Load	5.26%				7.89%			
Node	A	B	C	D	A	B	C	D
Data Queue Mean Delay	0.086	0.057	0.053	0.086	0.07	0.05	0.05	0.07
Node Utilization	0.021	0.02	0.021	0.02	0.03	0.03	0.03	0.03
Performance (non-normalized)	0.244	0.350	0.390	0.320	0.429	0.600	0.600	0.429
Performance (normalized) x 100	3.21	4.61	5.21	3.06	5.64	7.89	7.89	5.64
Arrival Rate Per Node [data_pk/sec]	0.5				1			
Per Cell Channel Offered Load	13.16%				26.32%			
Node	A	B	C	D	A	B	C	D
Data Queue Mean Delay	0.11	0.067	0.11	0.18	0.68	0.18	0.18	0.4
Node Utilization	0.052	0.054	0.054	0.056	0.11	0.098	0.1	0.1
Performance (non-normalized)	0.473	0.806	0.491	0.311	0.162	0.544	0.556	0.250
Performance (normalized) x 100	6.22	10.60	6.46	4.09	2.13	7.16	7.31	3.29
Arrival Rate Per Node [data_pk/sec]	1.5				2			
Per Cell Channel Offered Load	39.47%				52.63%			
Node	A	B	C	D	A	B	C	D
Data Queue Mean Delay	30	0.32	80	30	66	0.44	140	90
Node Utilization	0.14	0.15	0.09	0.14	0.14	0.2	0.06	0.14
Performance (non-normalized)	0.005	0.469	0.001	0.005	0.002	0.455	0.000	0.002
Performance (normalized) x 100	0.06	6.17	0.01	0.06	0.03	5.98	0.01	0.02

Table 13. All-in-one Case Simulation Results

8. Summary

Hidden-exposed node/cell performance measure results are not quite different from load case results for similar offered channel loads (channel here is the "common channel"). The worst cases observed are the "hidden node/cell" cases.

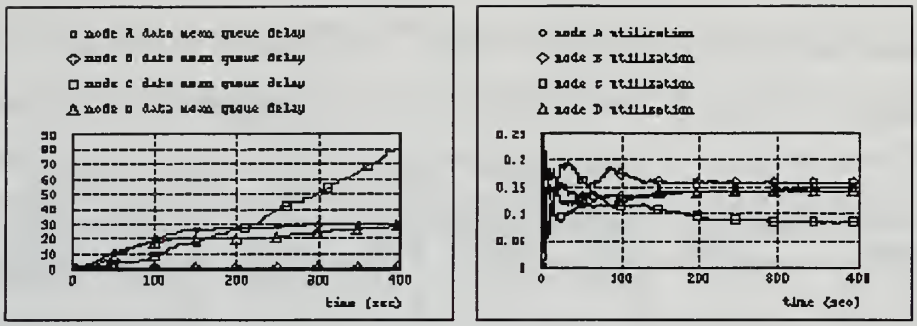


Figure 18. All-in-one Case Mean Delay and Utilization Graphs for 40% of Cell Load

Consider the following scenario for a "hidden cell network" (similar scenario is also applicable to "hidden node" topology) (Figure 19). Node A sends an RTS to B and B response with a CTS. While C is overhearing the CTS packet from B, node D sends an RTS to C so these packets collide and C receives/understands neither of them. when A receives B's CTS response it sends DS an DATA packets immediately. Since D could not get a response to its previous RTS to C it reschedules a new RTS (because it has data to send). If C receives this new RTS from D while A is sending its data to B, C's response (CTS) and A's DATA will collide on B's side and corrupt the DATA packet but D will successfully get the CTS response from C and will send DS and DATA packets. Since B has no idea that C is receiving a packet (it has missed node C's CTS response) it will diligently gave immediate response to any RTS requests which will be generated by node A. This of course will corrupt the DATA packet that C is being received.

Among the hidden-exposed node/cell cases "exposed cell" case has the best performance measure. This is due to the low backoff values of the receivers in the network. This is because there is no congestion at the receiver side at all.

9. Chapter Summary

During the simulations it is observed that the persistence of the transmitting nodes increase the performance of the network (at least up to four transmitting nodes). Persistence is related with the remote backoff values of the receivers in the transmitter. But this behavior needs to be investigated more for many number of transmitting nodes.

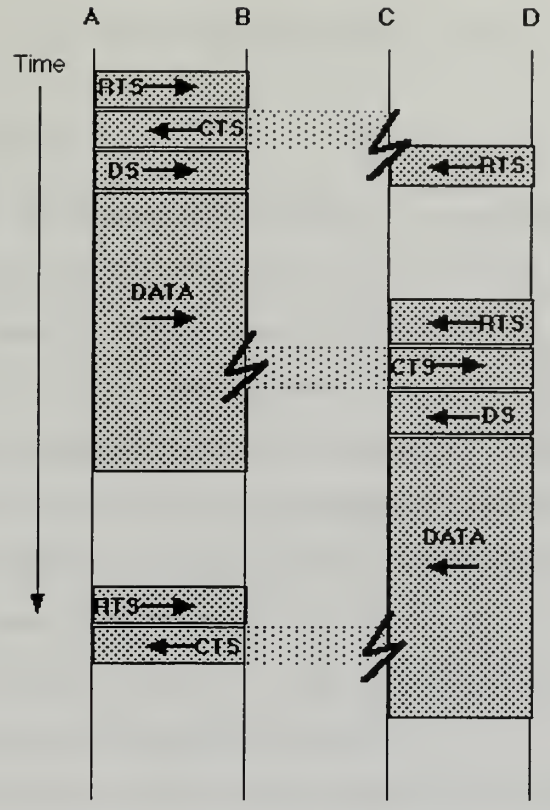


Figure 19. A Hidden-cell Network Communication Scenario

For all simulations, performance of the network dramatically decreases above 50% of common channel load, because of the high mean delay times. Maximum utilization observed is 50% (only base case and two cross communicating nodes cases has higher utilizations, base case has a maximum utilization of 66%) for simulated networks. Maximum performance measures are at 20-30% offered common channel loads of 3 and 4 transmitting nodes cases (both in load and hidden and exposed node/cell cases)

In the cases where backoff values in the network are "independent" from each other (as in one-way transmitting nodes and exposed node/cell cases) fairness is lost for high loads (above 60%). In these cases every transmitting node is communicating with a different node in the network. Although all nodes keep track of the backoff values of the neighboring nodes, these values do not effect each other at all. Thus, while one node in the network has a very small backoff value, other nodes can have very large backoff values for the same offered channel load.

VI. PROPOSED MODIFICATIONS

A. MODIFICATIONS

To improve the performance of the protocol, two modifications are thought to be useful and applied one at a time to the protocol. The same set of simulations are done for medium and high common channel load cases (from 40% to 90%).

From the results of the original protocol, it was observed that the persistence increases the network performance. So the first modification is to increase the persistence of the transmitting nodes. This is achieved by decreasing the backoff value multiplicand (K) from 1. 5 to 1.25. This provides a gentler/slower increase for remote backoff values in the transmitter and makes them more persistent.

The second modification applied to the original protocol is to provide carrier sensing (not collision detection) capability to the nodes.

In the original protocol, a node cannot understand that it is receiving a packet before all of the packet is received by the receiver module (at the physical layer) and passed to the MAC layer (where MACAW protocol runs). The probability of collision of two transmitting nodes is the probability of transmission of one node without recognizing (or before recognizing) that another node is transmitting. The transmission probability of a node when it assumes that the channel is available is based on the distribution function and its parameter value (in this case Poisson distribution with arrival rate value). The "recognition" period in the original MACAW protocol is the time that it takes for control packet transmission plus propagation delay. (time_slot+propagation_delay). Thus, we have a sample space of time_slot+propagation_delay for a transmission event that may cause a collision. When we introduce carrier sensing to the protocol, this sample space shrinks to one propagation_delay. Since our network design is based on the fact that the propagation delay is much less than a time slot (Chapter II, Protocol Overview) carrier sensing will decrease the probability of collisions by a factor of time_slot/propagation_delay.

Also consider the following scenario in a "hidden node" case (Figure 20) where A and C cannot hear each other but B can hear both. At time T A transmits an RTS to B. B gets it and responds with a CTS packet. Just before C completely receives/overhears and understands the packet it sends an RTS to B as well (since it is not aware of the fact that it is receiving a packet). This transmission causes C not to receive B's CTS response and collides with A's DS packet to B which is always followed by a DATA packet. So B will not be able to receive the DATA from A and C will not get a response for its RTS.

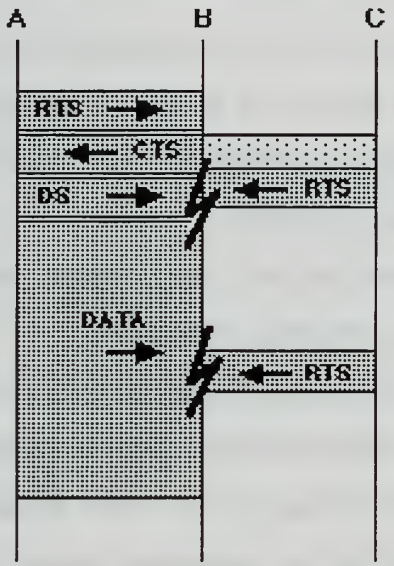


Figure 20. A Hidden Node Communication Scenario

B. SIMULATION RESULT DIFFERENCES

The behavior of the modified protocols is the same as the original protocol but some performance gains (Figure 21) and dependencies are observed. The summary of these differences are as follows.

1. For Load Cases

For the most of the test cases, decreasing the K value from 1.5 to 1.25 gave twice as good performance in the average than the original protocol results. It is also observed that the performance gain acquired by this modification gradually decreases as the number of the transmitting nodes increases regardless of the offered load (Figure 22).

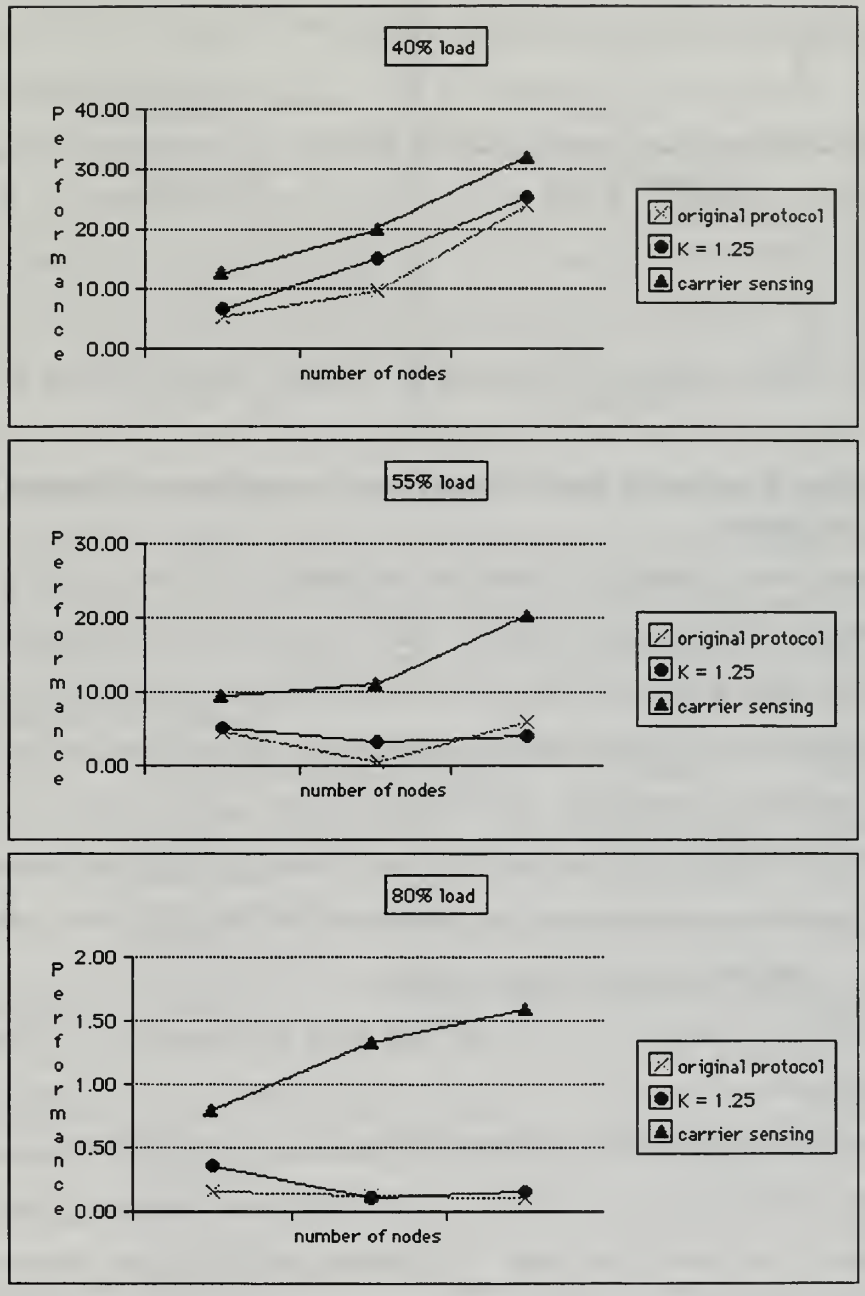


Figure 21. Performance Versus Number of Node Graphs for Different Loads

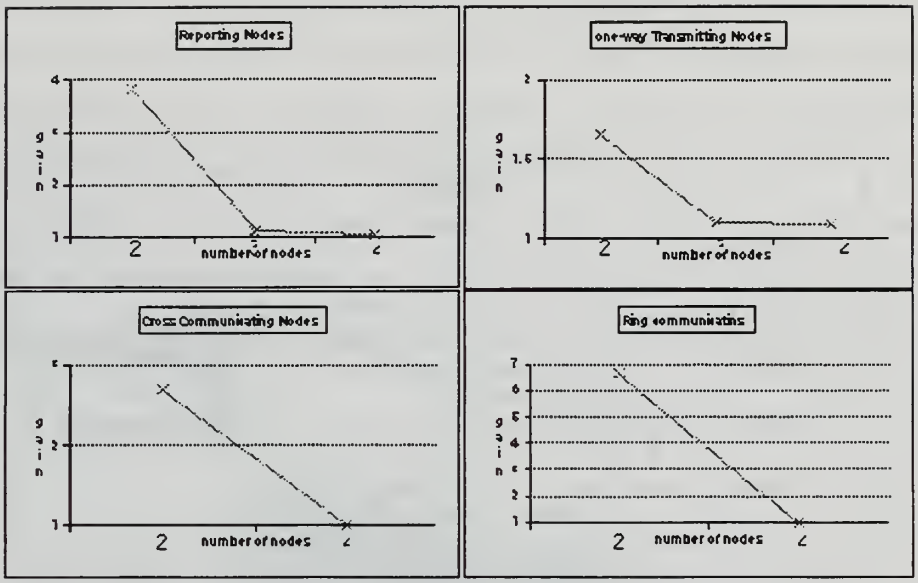


Figure 22. Gain Acquired by K=1.25 Modification versus Number of Nodes Graphs for Different Topologies

Carrier sensing capability increased the performance of the protocol in an average of 7 times for medium and high loads (40-80%). This is due to the lower mean delay times and 5-20% utilization gains in high loads (60-80%). The original protocol already provides maximum probable utilizations up to 60% of offered channel loads. Having carrier sensing increases the number of successful transmissions and keeps the remote backoff values in the transmitters low. The result is higher utilizations and lower delay times. This makes the original protocol's sharp performance decrease load threshold smoother and moves it from 50% to 60%.

2. Hidden-exposed Node/Cell Cases

Decreasing K from 1.5 to 1.25 had little effect on these cases. Performance gain is rather small or none.

Carrier sensing increased the performance more than previous modification. Significant performance measure increases observed for 50% and above common channel loads. The source of gain is lower mean delay times. While highest performance gain is achieved in hidden node case (for 50% offered load), no change was observed in the hidden cell case. In hidden cell case carrier sensing provides no help to the network to improve its performance, because

performance, because transmitters are out of range from each other (the only time that a transmitter utilizes carrier sensing in this topology is to sense its receiver's RRTS packet).

Another improvement achieved by carrier sensing is fairer channel access of the transmitting nodes (comparing to the original network) in the "hidden-exposed cell" and "all-in-one" network topologies.

In "hidden-exposed cell" network node C cannot get the common channel as frequently as in the original protocol. Now C should wait B to send all its packet and act accordingly (node C must understand what B is "saying"). But this increases the performance of node A slightly, because backoff values are dominant for the performance and here there is a big difference between congestion levels of the receivers (especially for high loads).

In "all-in-one" case fairness is achieved between inner nodes. In the original protocol for high loads (80%), one of the inner nodes has much higher performance measure than the other. Now with carrier sensing both have similar performances. The performance difference between inner and outer nodes are still great because of the congestion levels at the receivers.

Exposed Node Case with	Common Channel Load	39.47%		52.63%		78.95%	
	Node	A	B	A	B	A	B
ORIGINAL PROTOCOL	Performance (normalized) x 100	4.29	4.21	4.39	5.26	0.16	0.08
K = 1.25	Performance (normalized) x 100	5.13	7.31	3.17	5.21	0.36	0.32
CARRIER SENSING	Performance (normalized) x 100	5.26	7.59	4.54	5.53	0.58	0.16
Hidden Node Case with	Common Channel Load	26.32%		39.47%		52.63%	
	Node	A	C	A	C	A	C
ORIGINAL PROTOCOL	Performance (normalized) x 100	5.26	5.26	1.86	2.19	0.17	0.20
K = 1.25	Performance (normalized) x 100	5.36	6.29	2.46	2.49	0.06	0.32
CARRIER SENSING	Performance (normalized) x 100	7.31	5.72	3.29	2.63	1.22	1.46

Table 14. Original and Modified Hidden-Exposed Network Case Performance

Hidden-exposed Node with	Common Channel Load	39.47%			59.21%		
	Node	A	B	C	A	B	C
ORIGINAL PROTOCOL	Performance (normalized) x 100	4.21	4.39	3.99	0.08	0.18	0.10
K = 1.25	Performance (normalized) x 100	4.39	4.54	5.48	0.10	0.25	0.11
CARRIER SENSING	Performance (normalized) x 100	4.39	5.06	4.24	1.79	2.63	2.11
Exposed Cell Case with	Common Channel Load	26.32%		52.63%		78.95%	
	Node	A	C	A	C	A	C
ORIGINAL PROTOCOL	Performance (normalized) x 100	10.12	8.27	4.05	6.14	0.32	0.49
K = 1.25	Performance (normalized) x 100	7.74	7.64	5.48	5.26	0.72	1.57
CARRIER SENSING	Performance (normalized) x 100	7.74	8.22	5.81	4.61	1.32	1.63
Hidden Cell Case with	Per Cell Channel Offered Load	13.16%		26.32%		39.47%	
	Node	A	D	A	D	A	D
ORIGINAL PROTOCOL	Performance (normalized) x 100	3.62	5.26	1.92	2.29	0.08	0.10
K = 1.25	Performance (normalized) x 100	2.63	2.63	1.39	1.39	0.06	0.10
CARRIER SENSING	Performance (normalized) x 100	1.64	1.88	2.06	3.45	0.09	0.12
Hidden-exposed Cell Case with	Per Cell Channel Offered Load	13.16%		26.32%		39.47%	
	Node	A	C	A	C	A	C
ORIGINAL PROTOCOL	Performance (normalized) x 100	3.45	11.96	0.35	8.13	0.02	6.78
K = 1.25	Performance (normalized) x 100	4.11	11.96	0.06	8.13	0.02	7.52
CARRIER SENSING	Performance (normalized) x 100	4.67	10.12	0.39	5.53	0.03	7.45
All-in-One Case with	Per Cell Channel Offered Load	13.16%				19.74%			
	Node	A	B	C	D	A	B	C	D
ORIGINAL PROTOCOL	Performance (normalized) x 100	2.13	7.16	7.31	3.29	0.06	6.17	0.01	0.06
K = 1.25	Performance (normalized) x 100	2.06	7.74	7.74	2.19	0.25	6.02	0.01	0.05
CARRIER SENSING	Performance (normalized) x 100	2.19	6.58	6.58	1.88	0.06	2.95	3.20	0.12

Table 14. Continued

C. CONCLUSIONS

To improve the performance of the protocol two modifications were applied in turn and their effects observed. The first modification was to decrease the backoff increase multiplicand (K). This increased the performance of the protocol for most cases, and a relation between the number of transmitting nodes and K is observed (but this needs further study). Up to a point persistence obtained by decreasing the value of K caused an increase in performance, but as the number of transmitting nodes increased this persistence became a degrading factor to the performance.

The real performance gain was observed when the carrier sensing was introduced to the protocol. The effect of carrier sensing was more significant for high offered channel loads (60% and above). Simulation results showed that the channel acquisition with carrier sensing is superior than the original MACAW protocol channel acquisition policy.

VII. SUMMARY

In this thesis, performance characteristics of a packet radio multiple access protocol (MACAW -Medium Access Collision Avoidance Wireless), proposed by Bharghavan et al [Ref. 2] was investigated. Also some modifications were proposed to the protocol and some tests were conducted with original and modified protocols for different operational conditions.

The MACAW protocol is based on Karn's MACA protocol [Ref. 3] with some differences/improvements. These differences are as follows:

1. Use of RTS-CTS-DS-DATA-ACK message exchange, instead of RTS-CTS-DATA message exchange sequence.
2. Per "stream" basis "multiplicative increase linear decrease" backoff algorithm, instead of per node basis "binary exponential" backoff algorithm.
3. Distribution of congestion information via a "backoff copying" schema.

To test MACAW protocol under different operational conditions, its simulation model was built by OPNET 2.4c from MIL3, Inc. Many network topologies were constructed to test the protocol. These test cases were categorized as "load" and "hidden-exposed node/cell" cases. Tests were done for different offered channel/cell loads for each network topology. During these tests, each node's data mean queue delay times and utilizations were observed. A performance measure was introduced to compare the test results with each other. The formula of this measure is as follows:

$$Performance\_measure = \frac{Utilization\_of\_the\_network}{data\_mean\_queue\_delay} \quad (9)$$

The first case tested was a network of one transmitter and one receiver. The results of this test case was used as a basis of comparison for other test case results. The maximum utilization observed was 66% and a rapid performance decrease was determined after 80% of offered channel load as expected. This performance decrease is because of the high data

queue mean delay times in the transmitter. Secondly, the load cases were tested with two, three and four transmitting nodes with different network topologies, such as

- Mutually communicating nodes
- One-way communicating nodes
- Reporting nodes
- Ring like data exchanging nodes

During simulations, the channel utilization almost never exceeded 50% (maximum utilization observed was 66%). This 50% utilization is achieved at the expense of high data mean queue delay times. Performance of the networks were not acceptable above 50% of offered channel loads due to the high delay values. For high loads (above 60%) fairness was lost in "one-way communicating nodes" networks.

In "reporting" and "ring" cases channel access was always fair but transmitters' data mean queue delay times were high, thus performance was poor. It was also observed that when the number of the transmitting nodes increased the channel performance increased for 20% of offered channel loads. This is because the chance that there is a node ready to transmit when the channel is available is high in the networks with many transmitting nodes.

After load case tests, hidden-exposed node/cell cases were generated and tests were conducted with various common channel and /or cell loads. In hidden node and cell cases poor performances were observed. This is due to the interference that the hidden node/cell causes. The detailed scenario is given in Chapter V, Summary section.

After completing the tests with original MACAW protocol two modifications were applied to the protocol one at a time and their effects were observed. These modifications were based on the findings of the tests done with the original protocol. They were

- Decreasing the backoff increase multiplicand
- Introducing carrier sensing to the protocol

The same set of tests were performed with each modified protocol.

During the simulations of these modified and original protocols, a relation between persistence (caused by backoff algorithm) and the number of transmitting nodes were observed. The degree of persistence can be arranged by the backoff increase multiplicand and increasing persistence can improve the performance of the network, but as the number of the transmitting nodes increases in the network, this becomes a degrading factor. It was also observed that channel acquisition with carrier sensing is superior to the original MACAW protocol channel acquisition policy.

Backoff is a measure of congestion at the location of the node in interest. Distributing its backoff information throughout the network provides fair access to this node, but has no use if other nodes (those who copied this information) do not communicate with this node (in one-way communicating nodes cases, fairness is lost above 50% of channel offered loads). Also a relation between the number of transmitting nodes and backoff increase multiplicand is observed. The gain acquired by changing this value from 1.5 to 1.25 is decreased as the number of the transmitting nodes in the network increased, regardless of the channel load. Simulations emphasized the importance of the backoff algorithm for the protocol. A through analysis of the backoff algorithm can be done and new policies like dynamic backoff increase multiplicand and collective backoff calculation for the channel can be introduced to the algorithm.

Topologies used during the simulations are to investigate the behavior of the protocol under different operational conditions. Although each of the networks can potentially be in use in real life, no explicit effort is performed to achieve this. Some real life examples can be chosen to simulate. Also all nodes in the simulated networks were fixed. The effects of mobilization to the protocol can also be investigated (OPNET 2.4c provides this utility).

APPENDIX. MACAW PROTOCOL SIMULATION VIA OPNET2.4C

In the body of the thesis, a general overview of OPNET 2.4c is given and simulation results are discussed but simulation itself is not mentioned at all. Here, some starting points are given for the reader who needs a better understanding of the MACAW protocol simulation process. A much more complete form of this Appendix is located at "http://www.cs.nps.navy.mil/misc/networking/OPNETTutor/TOC.html".

OPNET is very sophisticated graphic oriented network design, simulation and analysis tool. To be able to use this tool, the reader should be familiar with the Unix environment and the C programming language.

OPNET has extensive documentation. These documents and their brief descriptions are as follows.

- Tutorial Manual 1.0: Introduction to OPNET with some simple example projects
- Modeling Manual 2.0: Detailed overview of OPNET from process design to analysis and built in object references
- MIL3 User Interface Manual 3.0: User interface issues
- Tool Operations Manual (Development) 4.0: OPNET Network editor, Node editor,Process editor, Parameter editor user guide
- Tool Operation Manual (Sim. & Analysis) 4.1: OPNET Probe editor, Simulation tool, Analysis tool, Filter editor user guide
- Simulation Kernel (anim-pk) 5.0: Simulation kernel function references from op_anim* to op_pk* inclusively
- Simulation Kernel (prg-topo) 5.1: Simulation kernel function references from op_prg* tp op_topo* and OPNET global constants
- External Interfaces Manual 6.0: System Administrator related information, command line simulation execution/debugging, accessing OPNET models externally

- MIL3 Utility Program Manual 7.0: Various utility programs of OPNET (map editor, orbit generator etc.)
- Example Models Manual (Base and General Models Vol.0) 8.0.0: Some example process models
- Example Models Manual (Protocol Models Vol.0) 8.1.0: Some example protocol models

A good starting point to learn OPNET is Tutorial Manual 1.0 and Modeling Manual 2.0.

Network design in OPNET can be done top-down or bottom-up. Either approach is applicable. A mixed approach is used during the implementation of the MACAW protocol. Shortly, first a node model that runs the MACAW protocol is built by using OPNET's built in modules, then the "process models" of programmable modules are generated and finally networks that uses this "MACAW node" is constructed.

The MACAW node model is built by OPNET Node Modeler. The node model has six modules

- data generator
- passive queue
- protocol processor
- radio transmitter
- radio receiver
- antenna

The first three of these modules are programmable, that is, each has an associated "process model." The job of these process models are as follows.

Data Generator Module Process Model: The job of this module is to create data packets for a given arrival rate with a Poisson distribution and pass them to the passive queue module. The processor will have two unforced states

- INIT
- ARRIVAL

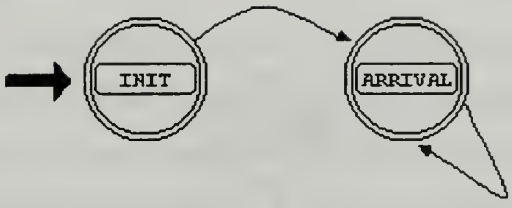


Figure 23. Data Generator Module Process Model Finite State Machine

When this process model is first invoked, INIT state enter executives are executed. There, "destination address" and "arrival rate" process attribute values are read and first data generation time is calculated according to Poisson distribution and a self interrupt is scheduled at that time. When this time is reached a self interrupt occurs and program flow goes to ARRIVAL state (there is no exit executives for INIT state). In this state a data packet is created and this packet is sent to passive queue module, then a new interrupt time is generated and scheduled. From now on this process stays in ARRIVAL state for all other interrupts.

Passive Queue Module Process Model: This queue module holds the packets until they are requested by the protocol processor. The process model of this module has three states

- BRANCH (forced)
- INS_TAIL (unforced)
- SEND_HEAD (unforced)

This process module responds to two types of interrupts "stream" and "access." "Stream interrupt" is generated when the data generator module sends a packet to this module.

When this happens, process goes to INS_TAIL state and inserts the packet at the end of the queue. "Access interrupt" is a polling from protocol processor module. When this interrupt occurs process goes to SEND_HEAD state and "quietly" sends the data packet at the head of the queue to the protocol processor. "quietly" means, without causing any interrupt at the destination module (which is protocol processor in this case).

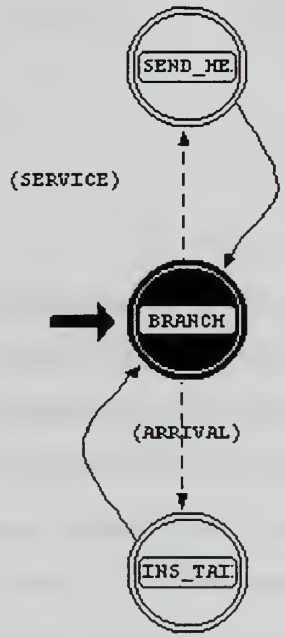


Figure 24. Passive Queue Module Process Finite State Machine

Protocol Processor Module Process Model: This process runs the MACAW protocol.The operation of the protocol is discussed in Chapter II (MACAW Protocol Summary) in detail. The finite state machine that should be drawn with OPNET process design tool as protocol processor is almost the same as the one in the MACAW Protocol Summary section of that chapter. The only differences are

- INIT state,
- "default" transitions

- actions associated with some transitions

INIT state is needed to initialize some state variables when this process is invoked for the first time. After that INIT state will never be reached.

The "default" transition is taken when no other transition condition is true. They are usually back to the same state.

After building the node model that runs MACAW protocol, networks that are used for test cases are constructed via OPNET's "Network Editor" and simulations are run.

There are two ways to run simulations in OPNET. First way is within OPNET working environment by using "Simulation" tool, the other way is invoking the simulation executable from Unix command line. Because of the number of the simulations, we preferred to run them from command line, this provides the usage of Unix's extensive scripting capability.

During the construction of the MACAW node, two variables are "promoted" as simulation parameters. These are

- generator.arrival rate
- generator.destination address

By using these variables different communication topologies with different channel loads (which is adjusted by arrival rates of transmitting nodes in the network) can be generated.

Finally results are analyzed by using OPNET's "Analysis Tool." A complete explanation of these steps are given in HTML documents (http://www.cs.nps.navy.mil/misc/networking/OPNETTutor/TOC.html)

LIST OF REFERENCES

1. Davis, P. T. and R. M. Craig, *Wireless Local Area Networks*, McGraw-Hill, Inc., 1995.

2. Bharghavan, V., A. Demers, S. Shenker and I. Zhang, "MACAW: A Media Access Protocol for Wireless LAN's," Proceedings of ACM SIGCOMM 94, ACM, 1994, pp. 212-25.

3. Karn, P. "MACA - A New Channel Access Method for Packet Radio," ARRL/CRRL Amateur Radio 9th Computer Networking Conference, ARRL, 1990, pp. 134-40.

4. Chane, F. and J. Garcia-Luna-Aceves, "Floor Acquisition Multiple Access (FAMA) for Packet-Radio Networks," Proceedings of ACM SIGCOMM 95, ACM, 1995, pp. 262-73.

5. Almquist, M. "Specification and Verification of a Wireless MAC Protocol," Master's Thesis, Naval Postgraduate School, September 1995.

6. Lundy, G., "Specification and Analysis of a Data Transfer Protocol Using Systems of Communicating Machines," Distributed Computing, 1991.

7. Sadiku, M.N.O. and M. Ilyas, *Simulation of Local Area Networks*, CRC Press, Inc., 1995.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center 8725 John J. Kingman Rd., STE 0944 Ft. Belvoir, VA 22060-6218	2
2.	Dudley Knox Library Naval Postgraduate School 411 Dyer Rd. Monterey, California 93943-5101	2
3.	Professor Gilbert M. Lundy Code CS/Ln Naval Postgraduate School Monterey, California 93943-5101	1
4.	Professor Man-Tak Shing Code CS/Sh Naval Postgraduate School Monterey, California 93943-5101	1
5.	Tufan Oruk 60'lilar Sitesi Dicle Apt. Blok 4 No:1 Aydinevler/Kucukyali, Istanbul/TURKEY	1
6.	Deniz Kuvvetleri Personel Daire Baskanligi Bakanliklar, Ankara/TURKEY	1
7.	Deniz Harp Okulu Komutanligi 81704 Tuzla, Istanbul/TURKEY	1